Piotr Rybka

Ekstrakcja danych w Pythonie

Wydawnictwo Naukowe Helion

Redaktor prowadzący: Małgorzata Kulik

Projekt okładki: Studio Gravite/Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki

Grafika na okładce została wykorzystana za zgodą AdobeStock.com.

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
helion.pl/user/opinie/eksdan
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-289-2169-6

Printed in Poland.

Spis treści

Od autora **13**

CZĘŚĆ I. PODSTAWOWE POJĘCIA **17**

ROZDZIAŁ 1. **Co można robić z danymi** **19**

1.1. Oczyszczanie 20
1.2. Normalizacja 21
1.3. Wzbogacanie 22
1.4. Agregacja 22
1.5. Kwerendowanie 23
1.6. Pozyskiwanie, zbieranie, gromadzenie 23
1.7. Odzyskiwanie 24
1.8. Eksploracja 24
1.9. „Zeskrobywanie” 24
1.10. Transformacja 24
1.11. Integracja 25
1.12. Wydobywanie 26
1.13. Wydobywanie danych z tekstów 27
1.14. Parsowanie 27

ROZDZIAŁ 2. **Ekstrakcja danych** **29**

2.1. Definicja 29
2.2. Etapy 29
2.3. ETL, ELT, migracje 30

ROZDZIAŁ 3. **Rodzaje danych** **31**

3.1. Zawartość danych 31
3.2. Struktura lub format danych 31
3.3. Użycie lub funkcja danych 32

ROZDZIAŁ 4. **Jednostki danych** **33**

4.1. Bit 33
4.2. Półbajt 33
4.3. Bajt 35
4.4. Przedrostki wielokrotności jednostek 35
4.5. Słowo (maszynowe) 36

4.6. Jednostki budowy tabeli bazodanowej ... 37
4.6.1. Wartości atomowe ... 37
4.6.2. Pola ... 37
4.6.3. Rekordy ... 37
4.6.4. Krotki ... 37
4.6.5. Encje ... 38
4.6.6. Atrybuty ... 38
4.6.7. Schemat danych ... 38
4.7. Kubit ... 38

ROZDZIAŁ 5. **Źródła danych ... 39**
5.1. Bazy danych ... 39
5.2. Hurtownie danych ... 40
5.3. Jeziora danych ... 40
5.4. Delta Lakes ... 41
5.5. Pliki płaskie ... 41
5.6. Interfejsy programowania aplikacji webowych ... 41
5.7. Arkusze kalkulacyjne ... 42
5.8. Źródła „zeskrobywalne” ... 42
5.9. Usługi chmurowe ... 42
5.10. Urządzenia Internetu Rzeczy ... 42

CZĘŚĆ II. PLIKI BINARNE I TEKSTOWE ... 43

ROZDZIAŁ 6. **Charakterystyka plików binarnych i tekstowych ... 45**

ROZDZIAŁ 7. **Przykłady plików binarnych ... 49**
7.1. Format .wave ... 49
7.2. Format .bmp ... 53

ROZDZIAŁ 8. **Sposoby osadzania danych binarnych w plikach tekstowych ... 57**
8.1. Problem niekompatybilności danych binarnych i tekstowych ... 57
8.2. Kodowanie Base64 ... 58
8.3. Kodowania Base16 i Base32 ... 62

ROZDZIAŁ 9. **Pliki binarne i tekstowe w Pythonie ... 63**
9.1. Listowanie plików ... 63
9.2. Strumienie ... 64
9.3. Tryby strumieni ... 65
9.4. Funkcje strumieniowe ... 67
9.5. Odróżnianie plików binarnych i tekstowych ... 68
9.6. Ciągi bitów ... 69
9.7. Odczyt plików binarnych i tekstowych ... 69
9.8. Odczyt metadanych pliku ... 73

CZĘŚĆ III. KODOWANIE TEKSTU 75

ROZDZIAŁ 10. **Systemy pozycyjne zapisu liczb 77**

10.1. Ogólna postać k-cyfrowej liczby 77
10.2. Podstawa systemu pozycyjnego 78
10.3. Rozwinięcie liczby w systemie o podstawie 10 78
10.4. Ogólne rozwinięcie k-cyfrowej liczby w systemie o podstawie p 79
10.5. Interpretacja liczb w systemach pozycyjnych 79
10.6. Rozpoznawanie systemu zapisu 81
10.7. Systemy pozycyjne o różnych podstawach 81
10.8. Niepozycyjne systemy zapisu liczb 82
10.9. Konwersje na system dziesiętny 83
10.10. Konwersja na system dwójkowy 84
10.11. Konwersja na system o podstawie p 87
10.12. System dwójkowy a szesnastkowy 90
10.13. Konwersja ułamków 91
10.14. Notacja naukowa 93

ROZDZIAŁ 11. **Systemy notacji w Pythonie 95**

ROZDZIAŁ 12. **Strony (tablice) kodowe 97**

12.1. Strona (tablica) kodowa vs kodowanie 97
12.2. Strona kodowa czy kodowanie 98
12.3. Strategie tworzenia tablic kodowych 99
12.4. ASCII 99
12.5. Tablice kodowe ISO i Windows 105
12.5.1. Zakres kodów i liczba bajtów wymagana do zakodowania znaku 105
12.5.2. Zawartość tablic kodowych ISO 106
12.5.3. Zawartość tablic kodowych Windows 107
12.5.4. ANSI 107
12.5.5. Zalety i wady tablic ISO i Windows 107
12.6. Unicode (Unikod) 108
12.6.1. Innowacyjność Unikodu 108
12.6.2. Kompatybilność z wcześniejszymi tablicami kodowymi 109
12.6.3. Standaryzacja 110
12.6.4. Budowa Unikodu 110
12.6.5. Zakres kodów i liczba bajtów wymagana do zakodowania znaku — zamkniętość i otwartość Unikodu 110
12.6.6. Zawartość Unikodu 113
12.6.7. Definicja znaku pisarskiego 122
12.6.8. Stosowane kodowania 123
12.6.9. Wady Unikodu 124
12.6.10. Zestawy i wyszukiwarki znaków Unikodu 124
12.6.11. Ekwiwalencja i normalizacja 125

12.7. Znaki niedrukowalne, białe i kody sterujące 126
12.7.1. Znak końca linijki 127
12.7.2. Kody (znaki) sterujące ASCII 128
12.7.3. Dodatkowe kody (znaki) sterujące w Unicode 129
12.7.4. Białe znaki stosowane w edytorach tekstu 129
12.8. Łączące znaki diakrytyczne 129
12.9. Semigrafika 130

ROZDZIAŁ 13. **Unicode w Pythonie 140**

ROZDZIAŁ 14. **Kodowania 142**
14.1. Strategie tworzenia systemów kodowania 142
14.2. Kolejność zapisu bajtów — endianness 142
14.2.1. Przyczyny różnej kolejności zapisu bajtów 142
14.2.2. Big Endian („grubokońcówkowość") 143
14.2.3. Little Endian („cienkokońcówkowość") 144
14.3. Kodowanie 8-bitowe 144
14.4. UTF-32 i UCS-4 145
14.5. UTF-16 i UCS-2 146
14.5.1. Ogólny algorytm kodowania w UTF-16 146
14.5.2. Kodowanie bezpośrednie w UTF-16 147
14.5.3. Kodowanie rozdzielcze 147
14.5.4. Przykłady kodowania rozdzielczego 148
14.5.5. Zalety i wady UTF-16 i UCS-2 148
14.6. UTF-8 149
14.6.1. Prefiksy bajtów w słowie maszynowym 149
14.6.2. Ramki binarne słów maszynowych 151
14.6.3. Algorytm kodowania w UTF-8 151
14.6.4. Algorytm dekodowania w UTF-8 152
14.6.5. Endianness w UTF-8 153
14.6.6. Zalety i wady UTF-8 153
14.7. BOM (Bite Order Mark) 155
14.8. Rozpoznawanie kodowania 156

ROZDZIAŁ 15. **Kodowanie tekstu w Pythonie 160**

CZĘŚĆ IV. WYRAŻENIA REGULARNE 165

ROZDZIAŁ 16. **Wstęp 167**
16.1. Definicja wyrażeń regularnych 167
16.2. Silniki wyrażeń regularnych 168

ROZDZIAŁ 17. **Budowa wyrażeń regularnych 169**
17.1. Metaznaki i literały 169
17.2. Kropka 170

ROZDZIAŁ 18. **Zbiory, zakresy i klasy znaków** 171
18.1. Zbiory i zakresy 171
18.2. Klasy znaków 172
18.3. Klasy znaków POSIX 172

ROZDZIAŁ 19. **Kwantyfikatory** 174

ROZDZIAŁ 20. **Grupy** 176
20.1. Grupowanie, przechwytywanie, odwołania 176
20.2. Alternatywa 177
20.3. Odwołania bezwzględne i względne 178
20.4. Napisy puste 179

ROZDZIAŁ 21. **Granice** 180
21.1. Metaznaki i metasekwencje konsumujące i niekonsumujące 180
21.2. Granice jednostek tekstu 180

ROZDZIAŁ 22. **Znaki Unicode** 182
22.1. Wskazywanie znaku kodem Unicode 182
22.2. Klasy znaków Unicode 182

ROZDZIAŁ 23. **Podstawianie** 189
23.1. Podstawianie liter i zmiana kasztowości 189
23.2. Matryca podstawiania 189

ROZDZIAŁ 24. **Asercje, wyrażenia warunkowe, definicje** 193
24.1. Warunki pozytywne i negatywne, retrospektywne i prospektywne 193
24.2. Wyrażenia warunkowe 195
24.3. Definicje 195

ROZDZIAŁ 25. **Opcje (modyfikatory, flagi) wyrażeń regularnych** 197
25.1. Ogólne opcje wyrażeń regularnych 197
25.2. Stosowanie opcji w różnych silnikach 198
25.3. Składanie opcji 199
25.4. Komentarze 200
25.5. Opcje silnika PCRE 200

ROZDZIAŁ 26. **Wyrażenia regularne w Pythonie** 201
26.1. Wyszukiwanie 203
26.1.1. Funkcja search 203
26.1.2. Funkcja match 203
26.1.3. Funkcja fullmatch 203
26.1.4. Funkcja findall 203
26.1.5. Funkcja finditer 204
26.2. Flagi (opcje) 204
26.3. Obiekt match object 205
26.3.1. Funkcja match_obj.group 205
26.3.2. Funkcja match_obj.groups 206
26.3.3. Funkcja match_obj.groupdict 206

26.3.4. Funkcje match_obj.start i match_obj.end 207
26.3.5. Funkcja match_obj.span 207
26.3.6. Atrybuty match_obj.lastindex i match_obj.lastgroup 207
26.3.7. Atrybuty pos i endpos, re i string 208
26.4. Inne funkcje 208
26.4.1. Funkcje sub i subn 208
26.4.2. Funkcja split 208
26.4.3. Funkcje re.escape i match_obj.expand 208
26.5. Przykłady użycia biblioteki re 209
26.5.1. Wyszukiwanie przy użyciu funkcji search, match i findall 209
26.5.2. Wyszukiwanie przy użyciu funkcji finditer 209
26.6. Podsumowanie 210

CZĘŚĆ V. FORMATY WYMIANY DANYCH 213

ROZDZIAŁ 27. **Wymiana danych 215**
27.1. Wymiana danych i schemat (struktura) danych 215
27.2. Serializacja i deserializacja danych 216

ROZDZIAŁ 28. **Języki znacznikowe 217**
28.1. Znacznik 217
28.2. Odmiany języków znacznikowych 217
28.3. Dane a metadane 218
28.4. Język a metajęzyk 219
28.5. Drzewa 220

ROZDZIAŁ 29. **Formaty CSV i TSV 222**
29.1. Budowa schematów CSV i TSV 222
29.2. Zagnieżdżenia w plikach CSV i TSV 222
29.3. Wskazywanie typów danych 223
29.4. Podsumowanie — zalety i wady 223

ROZDZIAŁ 30. **Pliki CSV w Pythonie 224**
30.1. Odczyt plików CSV 224
30.2. Zapis plików CSV 225
30.3. Parametry pliku i dialekty 226

ROZDZIAŁ 31. **Format JSON 230**
31.1. Dane, obiekty, tablice 230
31.2. Typy danych 230
31.3. Tablice (listy) 231
31.4. Liczby 232
31.5. Komentarze 232
31.6. Podsumowanie — wady i zalety 232

ROZDZIAŁ 32. **Pliki JSON w Pythonie 234**
32.1. Odczyt plików JSON 234
32.2. Zapis plików JSON 235

ROZDZIAŁ 33. **Format YAML** 237
33.1. Pary klucz-wartość 237
33.2. Komentarze 237
33.3. Typy danych 238
33.4. Listy 239
33.5. Obiekty 239
33.6. Znaki (sekwencje) ucieczki 239
33.7. Znaki Unikodu 240
33.8. Dodatkowe oznaczenia 240
33.9. Zapis czasu i dat – standard ISO 8601 241
33.10. Spacja po przecinku i dwukropku 243
33.11. Kotwice 244
33.12. Dyrektywy 245
33.13. Przykład dokumentu 245
33.14. Podsumowanie – zalety i wady formatu YAML 246

ROZDZIAŁ 34. **Pliki YAML w Pythonie** 247
34.1. Odczyt plików YAML 247
34.2. Zapis plików YAML 248
34.3. Własny parser 248

ROZDZIAŁ 35. **Format XML** 251
35.1. Rozszerzalność XML-a 251
35.2. Główne składniki dokumentu XML 252
35.3. Budowa elementu 252
35.4. Element czy atrybut? 253
35.5. Deklaracja XML 255
35.6. Deklaracje podstawowe 255
35.7. Instrukcje przetwarzania 256
35.8. Jednostki (encje) 257
35.9. Przestrzenie nazw 259
35.10. Atrybuty predefiniowane 261

ROZDZIAŁ 36. **Pliki XML w Pythonie** 263
36.1. Odczyt plików XML 263
36.2. Modyfikacja i zapis plików XML 264
36.3. Konstruowanie dokumentu 265
36.4. Konwersja na format XML 266

CZĘŚĆ VI. PRZESZUKIWANIE DOKUMENTÓW ZNACZNIKOWYCH 271

ROZDZIAŁ 37. **Gramatyki parsujące i gramatyki formalne** 273
37.1. Gramatyki formalne 273
37.2. Gramatyka w standardzie EBNF 275
37.3. Gramatyka parsująca w wyrażeniu regularnym 278
37.4. Użycie gramatyk parsujących w Pythonie 280
37.4.1. Gramatyka w wyrażeniu regularnym 281
37.4.2. Gramatyka w notacji EBNF 282

ROZDZIAŁ 38. **JSON Pointer** **287**
38.1. Składnia wskaźników JSON Pointer 287
38.2. JSON Pointer w Pythonie 288

ROZDZIAŁ 39. **JSON Path** **291**
39.1. Składnia ścieżek JSON Path 291
39.2. JSON Path w Pythonie 294

ROZDZIAŁ 40. **XPath** **297**
40.1. Wersje składni XPath 298
40.2. Struktura ścieżek XPath 298
40.3. Pomijanie elementów 301
40.4. Predykaty 301
40.5. Łączenie wyników 306
40.6. Operatory logiczne 306
40.7. Operatory arytmetyczne 308
40.8. Funkcje napisowe 308
40.9. Funkcje agregujące 311
40.10. Wyodrębnianie elementów struktury węzła 312
40.11. Funkcje konwertujące 314
40.12. XPath w Pythonie 315

ROZDZIAŁ 41. **XQuery** **318**
41.1. Wyrażenia FLWOR 319
41.2. Dodatkowe operatory porównania 319
41.3. Instrukcje warunkowe 320
41.4. Funkcje użytkownika 320

CZĘŚĆ VII SCHEMATY FORMATÓW WYMIANY DANYCH **321**

ROZDZIAŁ 42. **JSON Schema** **323**
42.1. Słowa kluczowe 324
42.1.1. Metajęzykowe słowa kluczowe 324
42.1.2. Wersja bazowego schematu i identyfikator własnego schematu ($schema, $id) 324
42.1.3. Schematy zewnętrzne i zagnieżdżone ($ref, $id, $defs) 325
42.1.4. Odwołania rekurencyjne 327
42.1.5. Zewnętrzne przestrzenie nazw ($vocabulary) 329
42.1.6. Wielokrotne użycie obiektów ($anchor) 330
42.1.7. Odwołania dynamiczne ($dynamicAnchor, $dynamicRef) 331
42.1.8. Komentarze ($comment) 332
42.2. Anotacje (title, description, default, examples, deprecated) 332
42.3. Ograniczenia 334
42.3.1. Specyfikacja typu (type) 334
42.3.2. Typ wyliczeniowy (enum) 338
42.3.3. Ograniczenia napisów (maxLength, minLength, pattern) 338
42.3.4. Ograniczenia liczb (multipleOf, minimum, exclusiveMinimum, maximum, exclusiveMaximum) 341

42.3.5. Ograniczenia tablic (items, maxItems, minItems, uniqueItems, contains) ... 341
42.3.6. Ograniczenia obiektów (maxProperties, minProperties, required, properties, patternProperties, additionalProperties, propertyNames, dependencies, dependentRequired, dependentSchemas) ... 342
42.3.7. Wartości stałe (const) ... 344
42.3.8. Ograniczenia danych binarnych osadzonych w tekście (contentMediaType, contentEncoding, contentSchema) ... 346
42.4. Operatory warunków (allOf, anyOf, oneOf) ... 348
42.5. Wyrażenia warunkowe (if, then, else) ... 348
42.6. Specyfikacja głównego elementu (korzenia) ... 350
42.7. Kolizje nazw ... 351
42.8. JSON Schema w Pythonie ... 351

ROZDZIAŁ 43. **XML Schema** ... **354**

43.1. Puste schematy ... 354
43.2. Przestrzeń nazw XML Schema ... 355
43.3. Własna przestrzeń nazw ... 356
43.4. Łączenie schematu z dokumentem XML ... 356
43.5. Typy elementów i atrybutów ... 357
43.5.1. Typy wbudowane ... 357
43.5.2. Typy proste ... 359
43.5.3. Typy złożone ... 359
43.6. Deklaracje zawartości elementów ... 360
43.6.1. Definicja elementu z treścią w typie wbudowanym bez atrybutów ... 362
43.6.2. Definicja elementu z treścią w typie wbudowanym z atrybutami ... 362
43.6.3. Element z podelementami bez atrybutów ... 363
43.6.4. Element z podelementami z atrybutami ... 363
43.6.5. Element pusty bez atrybutów ... 364
43.6.6. Element pusty z atrybutami ... 364
43.6.7. Element z treścią w typie zmodyfikowanym bez atrybutów ... 364
43.6.8. Element z treścią w typie zmodyfikowanym z atrybutami ... 365
43.6.9. Element z treścią mieszaną bez atrybutów ... 365
43.6.10. Element z treścią mieszaną z atrybutami ... 366
43.7. Ograniczenia związane z elementami i atrybutami ... 366
43.7.1. Aspekty ... 366
43.7.2. Łączenie aspektów ... 367
43.7.3. Blokowanie wartości aspektów ... 368
43.7.4. Liczba wystąpień elementu ... 369
43.7.5. Opcjonalność atrybutu ... 369
43.7.6. Treści i wartości domyślne i stałe ... 370
43.7.7. Elementy i atrybuty nieokreślone ... 370

43.8. Wyprowadzanie typów 371
43.8.1. Blokady wyprowadzeń 372
43.8.2. Typy abstrakcyjne 372
43.8.3. Deklaracje globalne i lokalne typów 373
43.9. Składacze 374
43.10. Inne składniki schematów 374
43.10.1. Odwołania 374
43.10.2. Grupy elementów i atrybutów 375
43.10.3. Listy i kombinacje 375
43.10.4. Elementy zastępcze 377
43.10.5. Klucze i wartości unikatowe 378
43.11. Automatyczne generowanie schematów XML Schema 381
43.11.1. XSD/XML Schema Generator 381
43.11.2. Visual Studio 382
43.12. Mapowanie dokumentu XML w programie MS Excel 382
43.13. Pliki XML w Excelu — importowanie i eksportowanie danych 385
43.14. XML Schema w Pythonie 385

Bibliografia 388

Skorowidz 398

Od autora

Ze względu na to, jak szybko wiedza przestaje być aktualna, omawianie jakichkolwiek zagadnień z zakresu informatyki jest trudne. Jeszcze trudniejsze jest nauczanie przedmiotów informatycznych, gdyż do dezaktualizacji wiedzy dochodzi to, że wszelkie przedmioty, czy to szkolne, czy akademickie, są w istocie różnego rodzaju propedeutykami, czyli wstępami do zagadnień lub ich zarysami. Jak zatem pisać podręczniki do przedmiotów informatycznych?

Miałem okazję przyjrzeć się edukacji informatycznej zarówno od strony studenta (w moich czasach szkolnych informatyka była dosłownie w powijakach), jak i wykładowcy, a że wówczas miałem jeszcze blisko 10-letnie doświadczenie w nauczaniu i układaniu programów nauczania, a nawet pisaniu podręczników, nieco łatwiej było mi odnaleźć się w roli prowadzącego zajęcia z komputerowej ekstrakcji danych, które przygotowałem dla studentów informatyki społecznej na AGH w Krakowie.

Nauczyciele akademiccy mają dużą dowolność w układaniu sylabusów zajęć. W przypadku przedmiotów informatycznych możliwe są w zasadzie dwa rozwiązania: albo przedstawiać najnowszy stan wybranej technologii (podejście wąskie), albo wybrać tematy jak najbardziej ogólne, uniwersalne, niezmienne (podejście szerokie). Wobec szybkiej dezaktualizacji wiedzy informatycznej i propedeutycznego charakteru większości przedmiotów szkolnych i akademickich, pierwsze podejście grozi tym, że po zakończeniu nauki uczniowie i studenci mogą mieć wiedzę niepotrzebną: za wąską, a w danej specjalizacji — zbyt ogólną i nieaktualną. Drugie podejście wcale nie jest lepsze, gdyż wiedza ogólna może być po prostu zbyt teoretyczna, co jej posiadaczom może utrudnić wybór konkretnego stanowiska. Istnieje też ryzyko, że na pewnym poziomie ogólności informatyka w zasadzie stanie się wykładem z matematyki, logiki, lingwistyki, cybernetyki, synergetyki lub jeszcze innej nauki.

Jeśli chodzi o ekstrakcję danych, której przyszło mi uczyć, miałem dodatkowo to szczęście, że w tamtym czasie zajmowałem się praktyczną ekstrakcją danych. Praca własna, a także doświadczenia innych osób pozwoliły mi podjąć decyzję o odrzuceniu dostępnych opracowań na temat ekstrakcji danych. Do dzisiaj nie jest ich zresztą wiele, a ich autorzy koncentrują się na bardzo wąskich przypadkach ekstrakcji. Prawie zawsze jest to *web scraping*. W swojej pracy zawodowej nigdy nie zajmowałem się pozyskiwaniem danych ze stron internetowych — akurat nie było mi to potrzebne. Natomiast przydała mi się znajomość wyrażeń regularnych, kodowania tekstu, budowy języków znacznikowych i sposobów przeszukiwania dokumentów zawierających znaczniki. Te doświadczenia pozwoliły mi znacznie zmodyfikować program powierzonych mi zajęć: rozszerzyć je o zagadnienia, które jeszcze długo się nie zdezaktualizują, a które są czasem nieznane nawet doświadczonym programistom.

Niniejsza książka to poprawiony, uzupełniony i poszerzony skrypt, który przygotowałem dla studentów prowadzonego przeze mnie laboratorium. Składa się z siedmiu części. Pierwsza zawiera omówienie podstawowych pojęć związanych w różny sposób z ekstrakcją danych. Wyliczenie kilkunastu możliwych działań na danych — co wcale nie wyczerpuje wszystkich możliwości — pozwala w rozdziale 2. nakreślić nie tyle granice ekstrakcji *sensu stricto*, ile najważniejsze cechy innych sposobów przetwarzania danych, które można stosować przy ich ekstrakcji. Równie szkicowe są klasyfikacje danych w rozdziale 3.: to zupełnie podstawowe podziały, najczęściej spotykane i wystarczające w większości przypadków. Jednostki danych podane w rozdziale 4. również nie wyczerpują tematu, ale mają za zadanie ułatwienie lektury kolejnych rozdziałów. I wreszcie rozdział 5., na temat źródeł danych, może być punktem odniesienia w poszukiwaniu konkretnych rozwiązań technicznych.

Pliki różnego rodzaju to jedno z najczęstszych źródeł danych. Ich podstawowy podział — pliki binarne i tekstowe — jest tematem części II. Obie grupy mają swoje wady i zalety, lecz co ważniejsze, wiążą się z zupełnie odmiennym podejściem do organizacji zawartych w nich danych. Nie są to oczywiste kwestie, dlatego ogólna charakterystyka w rozdziale szóstym jest wsparta analizą dwóch prostych formatów binarnych: *.wave* i *.bmp*. Wybranie pliku dźwiękowego i graficznego było celowe: ukazuje konieczność rozumienia specyficznych cech samych danych, zanim przystąpi się do studiowania specyfikacji danego formatu. Jeśli bowiem nie rozumie się pojęć takich jak głębia bitowa lub częstotliwość próbkowania, tym bardziej niezrozumiałe mogą być specyfikacje konkretnego formatu.

Cechą szczególną plików tekstowych jest ich kodowanie, ale temu poświęcona jest część III. Wcześniejszą zamykają zagadnienia niekompatybilności danych binarnych i tekstowych oraz sposoby osadzania tych pierwszych w plikach zasadniczo tekstowych. Te ostatnie bowiem mają pewne ograniczenia, co powoduje, że przenoszenie danych binarnych do pliku tekstowego jest problemem, ale nie odwrotnie: pliki tekstowe w zasadzie też są binarne, gdyż mają właśnie postać ciągu liczb.

Omówienie kodowania tekstu zaczyna się od przypomnienia wiadomości z zakresu systemów zapisu liczb. Nie jest to zagadnienie związane bezpośrednio z ekstrakcją tekstu, lecz odświeżenie tej wiedzy może ułatwić zrozumienie operacji na bitach i bajtach w rozdziale 14. Zawarty w nim opis najważniejszych standardów kodowania ilustruje dość długą drogę, jaką trzeba było przebyć, by opracować powszechne dzisiaj kodowanie UTF-8. Nie jest ono idealne, ale lepsze od wcześniejszych projektów.

Samo kodowanie jednak to tylko jedna strona medalu. Druga to zamiana znaku pisarskiego na liczbę. A tu kryją się dwie zagadki: co zamieniać (a więc czym jest znak) i jak zamieniać. Problematykę tablic kodowych trzeba omawiać przed kodowaniami, dlatego zrobimy to w rozdziale 12. Rozdział ten w zasadzie obejmuje także rys historyczny: od najprostszych rozwiązań do Unikodu — obecnie optymalnego.

Część IV poświęcona jest wyrażeniom regularnym. To w zasadzie metajęzyk opisywania gramatyk generowania tekstów, który zaczęto wykorzystywać w innych celach. Zasady użycia wyrażeń regularnych składają się w istocie z dwóch głównych części: jądra, identycznego w niemalże każdym dialekcie, oraz metaznaków i metasekwencji, które nie tylko mają różną postać w różnych dialektach, ale nawet mogą być w nich nieobecne.

Warto jednak wiedzieć o ich istnieniu oraz o tym, że mogą nie być dostępne w używanej odmianie wyrażeń regularnych.

Kolejna, piąta część kontynuuje wątek plików tekstowych. Ta problematyka nie kończy się bowiem na kodowaniu. Podobnie jak pliki binarne, również pliki tekstowe mogą mieć swoistą strukturę. Mamy tu dwa rozwiązania: formaty tabelaryczne (CSV, TSV) i oparte na strukturze drzewa (JSON, YAML, XML). Opisane w rozdziałach 29., 31., 33. I 35. formaty są w istocie metajęzykami: to bardzo ogólne zasady tworzenia języków do konkretnych zastosowań. Wyjaśnienie tego rozróżnienia jest zawarte w rozdziale 28. Jest ono teoretyczne, ale warto poświęcić czas na jego zrozumienie, gdyż CSV, JSON lub XML to nie są konkretne języki, ale metody budowania własnych języków pozwalających na wymianę danych o danym kształcie. A to jest zagadnienie związane z serializacją i deserializacją, którym poświęcony jest rozdział 27. Nie mniej istotne jest objaśnienie koncepcji znacznika w rozdziale 28., które pozwoli lepiej zrozumieć budowę formatów drzewiastych.

Przeszukiwanie plików, czy to o strukturze tabeli, czy drzewa, może być wykonywane na wiele sposobów. Jednym z nich są gramatyki zakodowane w formie wyrażeń regularnych. Inne metody, szczególnie przydatne w odniesieniu do struktur drzewiastych, są omówione w rozdziałach od 37. do 41.

Schematy formatów wymiany danych objaśnione w ostatniej, siódmej części stanowią metaopis konkretnych języków znacznikowych. Ograniczyłem się do tych najbardziej złożonych: schematów JSON-owych (rozdział 42.) oraz XML-owych (rozdział 43.).

Teoretycznemu omówieniu każdej technologii towarzyszy rozdział poświęcony użyciu tej technologii w Pythonie. Podane przykłady powinny działać w wersjach Pythona od 3.7 wzwyż. Wybór akurat tego języka programowania nie jest przypadkowy: mimo dużej prostoty, a przez to łatwości opanowania, język ten ma duże możliwości, dlatego często się go stosuje właśnie do przetwarzania danych. Najprostszy program typu „Hello world" napisany w Pythonie zawiera tylko jedną linijkę kodu, nie trzeba zatem pisać dużej ilości kodu, by osiągnąć najprostszy rezultat. Z tego powodu Python jest idealnym językiem do pisania skryptów — krótkich programów wykonujących dość proste zadania. Takim zadaniem może być właśnie ekstrakcja danych.

Zawartość tej książki z całą pewnością nie wyczerpuje zagadnienia ekstrakcji danych. Jednakże liczba możliwych przypadków ekstrahowania danych jest tak duża, że można by dopisywać rozdziały w nieskończoność. Każdy kolejny byłby jednak studium konkretnego, bardziej szczegółowego przypadku. Proponowany zestaw opracowałem z myślą o jak najogólniejszym podejściu do zagadnienia, a konkretne rozwiązania — takie jak formaty wymiany danych lub sposoby przeszukiwania dokumentów znacznikowych — przywoływałem jedynie wtedy, gdy można w ich przypadku z dużą pewnością założyć, że nie zostaną zbyt szybko zastąpione innymi. Zapewne jest więcej technologii, które raczej będą nam jeszcze długo towarzyszyć lub które można omówić na pewnym poziomie ogólności, tak aby ukazać generalne zasady pozyskiwania danych. Niewykluczone zatem, że kolejne wydania tego opracowania zostaną uzupełnione lub zmodyfikowane.

CZĘŚĆ I

PODSTAWOWE POJĘCIA

ROZDZIAŁ 1.

Co można robić z danymi

Na początku warto określić zakres tego, co nazywamy tu „ekstrakcją danych". Przyjrzyjmy się wymienionym na poniższej liście definicjom terminów, z których wiele ma zbliżone znaczenie, co pozwoli zrozumieć, jak ma się ekstrakcja danych do innych możliwych czynności związanych z danymi.

Oprócz ekstrakcji danych (ang. *data extraction*) możemy zatem mówić o:

- oczyszczaniu danych (ang. *data cleansing*),
- eksploracji danych (ang. *data exploration*),
- wydobywaniu danych (ang. *data mining*), w tym z tekstów (ang. *text mining*),
- „zeskrobywaniu"[1] danych (ang. *data scraping*),
- transformacji danych (ang. *data transformation*, *data wrangling*, *data reshaping*),
- zbieraniu danych (ang. *data harvesting*),
- integracji danych (ang. *data integration*),
- gromadzeniu danych (ang. *data collecting*),
- pozyskiwaniu danych (ang. *data retrieval*),
- odzyskiwaniu danych (ang. *data recovery*),
- kwerendowaniu danych (ang. *data querying*),
- normalizacji danych (ang. *data normalization*),
- agregacji danych (ang. *data aggregation*),
- wzbogacaniu danych (ang. *data enrichment*),
- parsowaniu danych (ang. *parsing data*).

Wiele z powyższych terminów dotyczy działań, które w pewnej mierze się pokrywają. Ich istotą jest jednak to, że wskazują na konkretny cel danego przetwarzania. Może on wiązać się z wykonywaniem innych czynności, mających inne cele, ale rozróżniając te procesy, będziemy się kierowali celem nadrzędnym.

[1] Zwykle nie tłumaczy się terminu *data scraping*, ale gdybyśmy chcieli podać polski odpowiednik, to „zeskrobywanie" byłoby chyba najlepszym.

1.1. Oczyszczanie

Jedną z najczęstszych czynności wykonywanych tuż po wyekstrahowaniu danych jest ich **oczyszczanie** (ang. *data cleansing*). Otrzymane dane mogą bowiem zawierać błędy, które w najlepszym razie uniemożliwią dalsze przetwarzanie, a w najgorszym znacznie pogorszą jakość wykonywanych analiz.

Oczyszczanie danych obejmuje zwykle:

- usunięcie zduplikowanych rekordów (wierszy) w tym samym źródle lub pochodzących z różnych źródeł (deduplikacja);
- opracowanie brakujących wartości — istnieje tu wiele możliwości, np.
  - imputacja — wstawianie sztucznych danych, zwykle obliczanych jako średnia, mediana, dominanta wartości rzeczywistych;
  - uzupełnianie wsteczne lub następcze, możliwe w przypadku danych chronologicznych, np. ciągu dat — na podstawie daty następnej lub poprzedniej próbuje się ustalić brakującą;
  - usuwanie rekordu — ostateczność w sytuacji, gdy nie da się odtworzyć przypuszczalnej wartości (rekord w takim przypadku jest i tak bezużyteczny, więc może być bezpiecznie usunięty);
- korekta ewidentnych błędów literowych, choćby wynikających z błędów rozpoznawania tekstu (OCR), np. zamiana 1 na l;
- walidacja, np. sprawdzanie, czy wprowadzony adres e-mailowy jest poprawnym adresem, czy dane nie wykraczają poza granice (np. ujemny wiek);
- rozwiązywanie konfliktów danych, np. wówczas, gdy dwa różne źródła zawierają różne informacje;
- konsolidacja — usunięcie pozornych nieduplikatów, ale w praktyce wartości zduplikowanych, tzn. różnych na pierwszy rzut oka, jednak w istocie identycznych, np. po ujednoliceniu jednostek lub formatu;
- usunięcie początkowych i końcowych białych znaków (jeśli nie wynika to z potrzeby ujednolicenia danych);
- usunięcie tzw. słów stopujących (ang. *stop words*) — zwykle wyrazów funkcyjnych (spójniki, przyimki, słowa posiłkowe, zaimki), które jedynie współtworzą gramatyczność tekstu, ale same w sobie nie wnoszą znaczeń (może to być czynność normalizująca, jeśli wynika z konieczności ujednolicenia danych);
- usunięcie tzw. *hapax legomena* — wyrazów lub związków wyrazowych występujących tylko raz w danym tekście i niespotykanych nigdzie indziej[2];
- usunięcie interpunkcji (może być również czynnością normalizującą).

[2] *Hapax legomena* utrudniają lematyzację, rdzeniowanie, tokenizację itp., gdyż żaden słownik ich nie zawiera, a ze względu na wyjątkowość przetwarzanie ich jest zwykle bezcelowe.

1.2. Normalizacja

Doprowadzanie danych do zgodności z założoną normą bywa nierzadko nazywane zamiennie oczyszczaniem. Różnica polega jednak na tym, że w przypadku **normalizacji** (ang. *data normalization*) musimy założyć pewien model lub standard danych, który pozwoli nam ujednolicić wszystkie dane. Jest to zatem coś bardziej zaawansowanego niż oczyszczanie, które polega na usuwaniu ewidentnych błędów. Nie muszę wiedzieć, jak powinna być zapisana data, by oczyścić rekord z wartością `ol-12-2o22` (powinno być `01-12-2022`), ale do celów normalizacyjnych muszę znać założony standard zapisu.

Podczas normalizacji zwykle nie usuwa się już rekordów, gdyż dane powinny już być czyste. Brakuje im jedynie ujednolicenia.

Typowe czynności normalizacyjne obejmują:

- ujednolicenie kasztowości: wielkie lub małe litery, formatowanie typu PascalCase, camelCase, snake_case itp.;
- ujednolicenie kodowania (np. wszystkie napisy w UTF-8);
- usunięcie interpunkcji (może być również czynnością oczyszczającą);
- usunięcie początkowych i końcowych białych znaków (jeśli nie jest to oczyszczaniem danych);
- konwersje typów wyliczeniowych, np. zamiana etykiet na wartości liczbowe lub odwrotnie;
- lematyzacja lub rdzeniowanie: zamiana wyrazów na ich formy hasłowe, tzw. lematy — wyrazy, którymi zaczynają się hasła słownikowe (np. polskie `kota`, `kotu`, `kocie` na `KOT`), lub na rdzenie — człony wyrazów po odcięciu wszystkich przyrostków i przedrostków sprowadzone do postaci sprzed wszelkich zmian fonetycznych (np. `gaś` zamiast `gasić`, `gaśnica`, `przygaszony`);
- tokenizacja — zamiana tekstu na tokeny, minimalne jednostki maszynowego przetwarzania tekstów, w dużym skrócie pojedyncze wyrazy, ale także znaki interpunkcyjne, czasem też końcówki ruchome (np. w polskim partykuła pytajna „-li” lub wzmacniająca „-że”);
- usunięcie tzw. *hapax legomena* — zob. wyżej;
- usunięcie tzw. słów stopujących — zob. wyżej;
- ujednolicenie zakresów, np. procenty mogą być zapisane dziesiętnie w zakresie [1, 0] (np. `0,12`) lub [100, 0] (np. `12%`);
- przetworzenie wartości skrajnych (ang. *outliers*), np. przez zastąpienie najbliższą akceptowalną wartością lub usunięcie rekordu zawierającego ekstremum;
- ujednolicenie zapisu daty i czasu (np. wszystkie daty i czas w standardzie ISO 8601: *`yyyy-mm-ddThh:mm:ss,sssssssss±hh:mm`*);
- parsowanie, np. rozłożenie daty na poszczególne komponenty (`12-01-2022` na `dzień: 12, miesiąc: 1, rok: 2022`);
- logarytmizacja — wyrażanie wartości w postaci jej logarytmu przy dowolnej, ale stałej podstawie, pomocne w przypadku dużego rozstępu danych, a w niektórych

innych (np. poziom natężenia akustycznego) praktycznie jedyny sposób reprezentowania wartości;
- dyskretyzacja — zamiana wartości ciągłych (nieprzeliczalnych) na dyskretne (przeliczalne), np. wagi osób na grupy osób o wadze w normie, z nadwagą lub niedowagą;
- ujednolicanie jednostek, np. metrycznych zamiast anglosaskich.

Jak widać, niektóre wymienione czynności były już wspomniane wcześniej w kontekście oczyszczania danych. Obie grupy zadań są bowiem dość zbliżone i tylko od tego, co uznamy za zabrudzenie danych, a co za brak jednolitości, będzie zależała granica między oczyszczaniem a normalizacją.

1.3. Wzbogacanie

Oczyszczaniu i normalizacji może towarzyszyć **wzbogacanie** (ang. *data enrichment*), które polega na uzupełnianiu danych o dodatkowe informacje, które można pobrać z innych źródeł lub wygenerować na podstawie danych oryginalnych. Oto kilka przykładów:
- dodanie współrzędnych geograficznych na podstawie adresu;
- powiązanie z innym informacjami, np. danych użytkowników z ich profilami w mediach społecznościowych, historią zakupów, preferencjami zakupowymi itp.;
- dane zagregowane, np. liczba kupionych przedmiotów, suma wydanych pieniędzy, suma wszystkich okresów, kiedy użytkownik był zalogowany;
- tzw. analiza sentymentu — badanie wypowiedzi pod kątem użytych wyrazów o nacechowaniu pozytywnym i negatywnym w celu ustalenia, jakie postawy przeważają (przychylne czy krytyczne);
- uzupełnienie danych, np. dodatkowego numeru telefonu lub adresu mailowego, adresu do korespondencji, imienia i nazwiska osoby kontaktowej itp.;
- proste obliczenia, np. czas od ostatniego zalogowania (data bieżąca minus data ostatniego zalogowania zamieniona na dni lub miesiące);

1.4. Agregacja

Agregacja (ang. data *aggregation*) może być elementem wzbogacania danych, ale zwykle dane zagregowane umieszcza się w osobnych tabelach lub widokach i stanowią one wynik dalszego przetwarzania danych, podczas gdy wzbogacanie jest etapem przygotowawczym i polega na uzupełnieniu istniejących tabel.

Agregowanie to określone łączenie danych (rekordów) w celu uzyskania nowych informacji. Może się pojawić na etapie przygotowania danych (ang. *preprocessing*), ale zwykle jest to już część analizy lub końcowych raportów.

Przykładowe agregacje:
- sumowanie wartości, np. suma wydanych pieniędzy, czas spędzony w grze;

- uśrednianie wartości, np. średnia ocen danego produktu lub sprzedawcy, mediana wynagrodzeń w danym sektorze gospodarki;
- obliczanie liczności, np. liczba odwiedzin strony internetowej, liczba postów na dany temat, liczba osób, które kupiły dany produkt;
- obliczanie wartości skrajnych — minimum i maksimum, np. najniższe i najwyższe temperatury zarejestrowane w kolejnych latach;
- obliczanie dominanty — najczęstszej wartości lub najczęstszego rekordu, np. najczęściej kupowany produkt w danej kategorii;
- obliczenia wagowe, np. średnia ważona ocen studentów — suma ocen pomnożonych przez ich wagi, którą następnie dzieli się przez sumę wszystkich wag każdej z ocen;
- sumy skumulowane — każda kolejna suma zawiera w sobie poprzednie wartości, tak że pierwsza suma jest równa pierwszej wartości, a ostatnia jest sumą wszystkich wartości (agregacja przydatna jest np. w monitorowaniu wzrostu sprzedaży lub czasu spędzonego przez zespół na realizacji zadania).

1.5. Kwerendowanie

Kwerendowanie (ang. *data querying*) to bodaj najprostsza czynność, jaką można wykonywać. Dane nie wymagają żadnych modyfikacji, są gotowe do odczytu, a kwerenda polega wyłącznie na wyborze tego, co jest nam potrzebne. Kwerenda jest więc w praktyce filtrowaniem i ewentualnie sortowaniem, ale nie ogranicza się jedynie do odczytu. Tworzenie (ang. *create*), aktualizowanie (ang. *update*) i usuwanie (ang. *delete*) też wchodzą w zakres kwerendowania. Dobrym przykładem są skrypty SQL-owe.

1.6. Pozyskiwanie, zbieranie, gromadzenie

Pozyskiwanie (ang. *data retrieval*) jest podobne do kwerendowania, ale dotyczy nie tyle bazy danych, ile całego systemu zarządzania bazą danych (ang. *Database Management System*). Systemy takie oferują bezpieczne (zarówno w sensie zewnętrznym — tylko osoby uprawnione mogą pozyskiwać dane, jak i wewnętrznym — zapytania nie powodują uszkodzenia danych) ich kwerendowanie, przy czym ograniczamy się tutaj tylko do odczytu danych.

Zbieranie, lub raczej „żniwa", danych (ang. *harvesting*) pojawia się wtedy, gdy mamy do czynienia z dużą ilością danych, a samo zbieranie wymaga dużych nakładów na automatyzację samego procesu.

Wreszcie **gromadzenie** (ang. *collecting*) kładzie akcent na dużą ilość zróżnicowanych źródeł informacji pobieranych zarówno automatyczne, jak i ręcznie. Dane mogą być różnego rodzaju i wymagać znacznego przetworzenia na wejściu (np. wprowadzenia do bazy danych ręcznie uzupełnionych ankiet zawierających pytania otwarte).

1.7. Odzyskiwanie

Jak sugeruje nazwa, w **odzyskiwaniu** (ang. *data recovery*) chodzi o wydobycie danych ze źródła, które nie jest łatwo dostępne wskutek uszkodzenia, wykasowania lub zagubienia. Najczęściej chodzi o odzyskanie danych z uszkodzonych nośników, takich jak dyski twarde lub pendrive'y. Jest to proces bardzo skomplikowany, gdyż każda ingerencja w nośnik może jeszcze bardziej uszkodzić dane i zupełnie uniemożliwić ich odzyskanie.

1.8. Eksploracja

W **eksploracji** (ang. *data exploration*) chodzi o wstępne zapoznanie się z danymi w celu ich ogólnego zrozumienia, tak aby można było odpowiedzieć na pytanie, o czym są te dane. Taka wiedzę trzeba mieć, by móc dalej przetwarzać dane.

W tym celu tworzy się bardzo podstawowe statystyki, takie jak liczba rekordów, średnia, mediana, dominanta, wartości skrajne, odchylenie standardowe itp. Pomocne mogą być również proste wykresy, np. histogramy, wykresy punktowe, liniowe, pudełkowe.

1.9. „Zeskrobywanie"

„**Zeskrobywanie**" (ang. *data scraping*) to specyficzny rodzaj ekstrakcji, w której dane nie są dostępne wprost i nie można ich bezpośrednio wykorzystać do dalszego przetwarzania. Mogą być i zwykle są zrozumiałe dla człowieka, ale dla analizatora automatycznego (programu, skryptu) są nieczytelne. Na wejściu mamy bowiem dane nieustrukturyzowane lub częściowo ustrukturyzowane, podczas gdy na wyjściu potrzebujemy danych ustrukturyzowanych (choćby częściowo, a najlepiej w całości).

Klasyczny przykład to *web scraping* — zeskrobywanie danych ze stron internetowych. Mimo że każda strona internetowa to plik w formacie HTML, który ma wyraźną strukturę, dane, których szukamy, są zwykle rozsiane po elementach i same mogą zawierać podelementy. Zeskrobywanie wymaga bardziej zaawansowanych i zautomatyzowanych czynności, takich jak parsowanie, oczyszczanie, normalizowanie, transformacja.

1.10. Transformacja

Jak sama nazwa wskazuje, **transformacja** polega na zmianie postaci danych. Może temu towarzyszyć zmiana struktury danych, a nawet wartości, lub może to być wyłącznie zmiana formatu wymiany danych.

Tak zdefiniowana transformacja danych (ang. *data transformation*, *data wrangling*, *data munging*, *data reshaping*) obejmuje w zasadzie wszelkiego rodzaju przetwarzanie danych,

gdyż zawsze dane na wejściu różnią się od danych na wyjściu. Ale transformacja jest zwykle jednym z etapów wczesnego przetwarzania (preprocessingu), a więc przygotowaniem danych do właściwej analizy. W tym ujęciu jednak dalej w skład transformacji powinno wejść oczyszczanie, normalizacja, a nawet agregacja. Różnica między nimi polega na tym, że oczyszczanie i normalizacja zazwyczaj są niezbędne, by można było cokolwiek zrobić z danymi, agregacja uzupełnia dane, podczas gdy transformacja jedynie ułatwia dalsze przetwarzanie.

Czasem wydziela się podtypy transformacji, zależnie od tego, jak dane zostały zmienione. Można tu wymienić trzy terminy:

- ***data reshaping***, zwany też *data munging* — zmienia się wyłącznie układ danych: dobór i kolejność kolumn i/lub dobór i kolejność wierszy;
- ***data wrangling*** — zmiana poziomu ustrukturyzowania danych: początkowe surowe dane otrzymują strukturę pozwalającą na dalsze przetwarzanie, czemu może towarzyszyć parsowanie, oczyszczanie, normalizacja, agregacja lub konsolidacja;
- ***data munging*** — jeśli już odróżnia się ten proces od *data wrangling*, to zwykle w celu oznaczenia doraźnych, niewielkich i ręcznych zmian w strukturze i zawartości danych.

1.11. Integracja

Istotą **integracji** (ang. *data integration*) jest łączenie danych (zwykle z różnych źródeł), które zawierają względnie zbliżony zakres informacji (jeśli jakieś źródło bardziej uzupełnia inne, mamy wtedy do czynienia ze wzbogacaniem).

Istnieje kilka metod integracji danych:

- **scalanie** (ang. *merging*) — źródła danych muszą mieć wspólne elementy, np. klucze (identyfikatory) lub kolumny pozwalające dopasować dane, dzięki czemu można np. zintegrować dane o zachowaniach klientów w różnych sklepach na podstawie ich danych osobowych; na wejściu mamy kilka źródeł X_i, każde ze zbliżoną liczbą danych n_i, a na wyjściu liczba rekordów jest równa max n_i;
- **konsolidacja** (ang. *consolidation*) — dane napływają po prostu z różnych źródeł, tak więc na wyjściu liczba rekordów wynosi $\Sigma\, n_i$;
- **mapowanie** (ang. *mapping*) — znajdowanie powiązań między danymi, np. na podstawie kluczy (identyfikatorów) lub wartości w określonych kolumnach (zwykle poprzedza scalanie);
- **integracja schematu** (ang. *schema integration*) — dopasowanie struktury danych z jednego lub wszystkich źródeł do wspólnego schematu, jest zatem wielokrotną transformacją danych.

1.12. Wydobywanie

Określenie „**wydobywanie danych**" (ang. *data mining*) zawiera trudne do oddania w tłumaczeniu odwołanie do wydobywania górniczego (ang. *mining* — 'wydobycie, górnictwo') i ono jest tu kluczowe. Zwykła ekstrakcja danych polega na pobieraniu informacji, które w dużej mierze są w danych obecne bezpośrednio. Wydobywanie idzie o krok dalej i próbuje „wycisnąć" z danych jeszcze więcej informacji. Za bardzo proste wydobywanie danych można by uznać agregację, ale w wydobywaniu stosuje się zwykle o wiele bardziej zaawansowane metody, które pozwalają odkryć niewidoczne na pierwszy rzut oka prawidłowości (ang. *patterns*) oraz zapewnić większe zrozumienie (ang. *insights*) analizowanych informacji.

Do najczęściej stosowanych metod wydobywania należą:

- **klasyfikacja** (ang. *classification*) — przypisywanie danych do wcześniej zdefiniowanych kategorii (klas) na podstawie zbliżonych cech, które są ekstrahowane ze zbioru uczącego — danych ręcznie podzielonych na klasy;
- **klastrowanie** (ang. *clustering*) — grupowanie danych na podstawie podanych wartości; w przeciwieństwie do klasyfikacji, grupy nie są znane na początku, a znajdowanie zgrupowań danych właśnie te grupy ukazuje;
- **regresja** (ang. *regression*) — określanie zależności między zmiennymi zależnymi a niezależnymi; w praktyce polega na określeniu parametrów określonego typu funkcji (liniowej — regresja liniowa, wielomianowej — regresja wielomianowa, wykładniczej — regresja wykładnicza i in.) na podstawie znanych wartości x i y, tak aby dla nowego x określić wartość zmiennej zależnej y; oprócz jednej zmiennej niezależnej można również zdefiniować funkcję o więcej niż jednej zmiennej (tzw. regresja wielokrotna, w przeciwieństwie do regresji prostej);
- **wykrywanie anomalii** (ang. *anomaly detection*) — wskazywanie nietypowych danych niepasujących do istniejącego schematu (pozwala znaleźć anomalie i rzadkie przypadki);
- **redukcja wymiarów** (ang. *dimensionality reduction*) — przydatna wówczas, gdy dane są opisane bardzo wieloma parametrami (np. mamy bardzo dużo kolumn danych) — pozwala uprościć dane i zmaksymalizować cechy znaczące;
- **analiza szeregów czasowych** (ang. *time series analysis*) — dotyczy danych przypisanych do konkretnych momentów w czasie, a więc szeregowalnych chronologicznie (np. od najwcześniejszych), i umożliwia zauważenie ogólnych trendów i okresowych zmian, a tym samym pozwala przewidywać ich przyszłe fluktuacje;
- **drzewa decyzyjne** (ang. *decision trees*), dzielące dane na mniejsze grupy na podstawie wartości, które najlepiej dzielą dane (uzyskana struktura drzewiasta może być użyta do klasyfikacji lub podejmowania decyzji);
- **sieci neuronowe** (ang. *neural networks*), czyli struktury wzorowane na ludzkich neuronach, a pozwalające na rozpoznawanie skomplikowanych wzorców (tekst, twarze i obiekty na zdjęciach lub nagraniach);
- **analiza przeżycia** (ang. *survival analysis*) — pozwala ustalić, w jakim czasie interesujące zjawisko się pojawi, co często stosuje się w medycynie do ustalenia, w jakim czasie zmieni się stan pacjenta zależnie np. od jego wieku i podanych leków.

1.13. Wydobywanie danych z tekstów

Ponieważ dane tekstowe nie poddają się tym samym analizom, co dane liczbowe, osobna grupa metod wydobywczych dotyczy właśnie analizy tekstów, inaczej **wydobywania danych z tekstów** (ang. *text mining*). Znajdziemy tu specyficzne metody pozwalające odnajdywać prawidłowości w tekstach. Spośród nich można wymienić m.in.

- **analizę sentymentu** (ang. *sentiment analysis*) — ocenianie nastawienia autora tekstu lub wypowiedzi, w praktyce przypisywanie tekstów do jednej z trzech kategorii: teksty pozytywne, negatywne i neutralne;
- **rozpoznawanie obiektów nazwanych** (ang. *named entity recognition*, NER) — wyszukiwanie w tekstach nazw osób, miejsc, organizacji itp., innymi słowy to wyszukiwanie nazw własnych, co jest przydatne nie tylko do oczyszczania tekstów, ale też do klasyfikowania tekstów na podstawie ich treści, a także generowania słów kluczowych;
- **modelowanie tematyczne** (ang. *topic modelling*) — określanie głównych tematów w tekście lub użytych w nim motywów;
- **analiza częstości** (ang. *frequency analysis*) wyrazów i grup wyrazowych (tzw. n-gramów — grup zawierających *n* wyrazów) — w praktyce budowanie list frekwencyjnych: list wyrazów z odpowiednio obliczoną częstością występowania (np. TF-IDF) i posegregowanych według tej częstości;
- **automatyczne streszczanie tekstów** (ang. *text summarizing*), dokonywane na dwa sposoby:
  - skracanie tekstu oryginalnego do najbardziej kluczowych zdań,
  - generowanie nowego tekstu, który zawiera treść tekstu wyjściowego;
- **osadzanie słów** (ang. *word embeddings*) — technikę tworzenia wektorowej reprezentacji słów, co pozwala ustalać ich znaczenie, kontekst i relacje semantyczne z innymi słowami, np. formułować równania typu wektor „król” + wektor „kobieta” = wektor „królowa”;
- **generowanie chmur wyrazów** (ang. *word cloud generation*), czyli bardzo prostej wizualizacji częstości użycia słów w tekście: najczęstsze wyrazy są największe i w centrum chmury, rzadsze nieco mniejsze, a najrzadsze są bardzo małe i na obrzeżach chmury (przygotowanie takiej chmury wymaga wcześniejszego wyczyszczenia tekstu z wyrazów stopujących oraz stworzenia list frekwencyjnych).

1.14. Parsowanie

Pojęcie **parsowania** (ang. *parsing*) ma w istocie wiele znaczeń, jednak w każdym z nich jest to podział ciągu pewnych elementów (znaków, wyrazów, wyrażeń…) na elementy (te same co poprzednio lub inne) powiązane ze sobą. Najprościej mówiąc, parsowanie to proces ustalania związków między elementami. Parsowanie tekstu lub napisu to innymi słowy jego analiza składniowa lub syntaktyczna (w przypadku danych tekstem jest plik tekstowy, np. w formacie JSON lub XML). Parsowaniem jest analiza zdań pojedynczych (rozpoznawanie zależności między podmiotem, orzeczeniem, dopełnieniem,

przydawką i okolicznikiem) oraz złożonych (zależności między zdaniami składowymi). Analiza składniowa odbywa się zawsze zgodnie z regułami określonej gramatyki, czyli regułami konstruowania (składania) wyrażeń z elementów danego języka.

Końcowym rezultatem parsowania zawsze jest **drzewo zależności**[3] (ang. *parse tree*), a więc rodzaj grafu, którego wierzchołki stanowią wydzielone elementy ciągu wejściowego, a krawędzie określają zależności między tymi elementami. Takie drzewo jest przede wszystkim strukturą logiczną, co oznacza, że nie zawsze wynik parsowania musi być przedstawiany jako graf. Deserializacja, zwłaszcza jeśli dotyczy formatu przechowującego struktury drzewiaste, wymaga zwykle wstępnego sparsowania ciągu wejściowego, a ponieważ dalszym etapem odczytu danych może być ich selekcja, to, co zostanie nam przedstawione, może w ogóle nie przypominać drzewa (np. gdy otrzymujemy w rezultacie listę liczb lub napisów). Struktura relacji sama w sobie może też być bardzo prosta (pojedyncza para klucz-wartość lub lista), co również utrudni dostrzeżenie struktury drzewiastej. Ale w istocie zawsze jest ona obecna, gdy mamy do czynienia z co najmniej parą powiązanych i niepodzielnych elementów.

Programy lub funkcje języków programowania wykonujące parsowanie nazywa się **parserami.**

[3] Zwane też drzewem wyprowadzenia, drzewem zależnościowym, derywacyjnym.

Skorowidz

A

ACID, 41
addytywny, 83
adres bazowy, 325, 327
adiustacja, 217
agregacja danych, 19, 22, 23, 25
akcent
  akutowy, 125
  grawisowy, 101, 014
akronim rekurencyjny, 237
aktualizowanie, 23
alias, 244, 245, 257–260, 320, 359
allow_nan, 235
alternatywa, 177–179
Amazon Redshift, 40
Amazon S3, 41, 42
American Standard Code for Information Exchange, 99
ampersand, 101, 102, 244, 257
amplituda, 52, 53
analityczne dane, 32
analityka w czasie rzeczywistym, 40
analiza
  częstości, 27
  przeżycia, 26
  sentymentu, 22, 27
analiza szeregów czasowych, 26
analytical data, 32
anchor, 244
angielski, 219
anomaly detection, 26
anotacje, 332, 333
anotowanie, 217
ANSI, 107
API, 41, 45
arkusz kalkulacyjny, 42, 347
array *patrz* tablica
ASCII, 51–55, 62, 69, 98–111, 128, 129, 133, 134, 144–154, 157–159, 172, 173, 182, 184, 198, 205, 235, 240, 253, 339, 340
asercje, 193
aspekt, 366–369
assertions, 193
atka, 100, 101
atom, 215
atomic data, 37, 38
atomowe dane, 37, 38
atrybut, 38, 201, 207, 208, 252–255, 258–262, 282, 354–383, 386
attr_type, 267, 269
attrib, 265, 266
attributes, 38
Audacity, 50, 51, 52
automatyczne streszczanie tekstów, 27
Avro, 215

B

bajt, 35
  młodszy, 34
  półbajt, 34
  starszy, 34
bajtowa, 147, 158
Base16, 58, 59, 62
Base32, 58, 59, 62
Base64, 58–62, 340
BaseX, 318
baza danych, 30, 39, 40
  grafowa, 40
  klucz-wartość, 40
  nierelacyjna, 40
  szeregów czasowych, 40
  tekstowa, 40
białe znaki, 126
big data, 40
Big Endian, 143–146, 153, 155
bin, 31, 95, 96, 160
binarna ósemka, 35
binarny, 48, 68, 80, 84, 97, 151, 152
binary digit, 33
binary eight, 35
Binary JSON, 215

bit, 33, 77, 100, 105, 147, 150, 152
młodszy, 33
starszy, 33
bit depth, 50
Bite Order Mark, 155
bitowa, 51, 53, 55
blok, 50, 52, 53, 124
nieznaków, 129
BMP, 53, 113–118, 131
bogaty, 31, 46, 48, 57
BOM, 155
BSON, 215
BufferedRandom, 69
BufferedReader, 69, 234, 247
BufferedWriter, 69
byte, 35
bytearray, 67–69
bytes, 67–69, 73, 266, 267
BZIP2, 347

C

C#, 100, 168, 199, 232, 259
camelCase, 21
canonical equivalence, 125
case sensitivity, 253
categorical data, 31
CD, 47, 50
CDATA, 256
character table, 97
check_circular, 235
chmura wyrazów, 27
chr, 140
chunks, 50
ciągi bitów, 69
ciśnienie akustyczne, 49
classification, 26
clean data,32
close, 64, 68
cloud storage services, 42
clustering, 26
code page, 97
collecting, 23
combining diacritical marks, 129
Comma Separated Values, *patrz* CSV
compatibility, 125
compile, 201, 202, 203, 209–211, 248, 282, 353
compositor, 374
computer file, 45
consolidation, 25
control characters/codes, 126
count, 202, 208, 311, 312
create, 23
Crockford Douglas, 230
CSV, 31, 215, 218, 222–232, 276, 278, 280, 282
cyfra, 46, 77, 78, 82, 93, 124, 151, 183
binarna, 33
szesnastkowa, 32
częstości, 27
częstotliwość próbkowania, 50, 52
częściowo ustrukturyzowane dane, 24, 31, 323
czterosłowo, 37
czytelne dla człowieka, 332

D

dane, 32, 218
agregacja, 19, 22
analityczne, 32
atomowe, 32
brudne, 32
czasowe, 31
częściowo ustrukturyzowane, 24, 31, 323
czyste, 32
czytelne dla człowieka, 32
czytelne dla maszyny, 32
deduplikacja, 20
eksploracja, 24
ekstrakcja, 29
geoprzestrzenne, 31
gromadzenie, 23
imputacja, 20
integracja schematu, 25
integracja, 25
kategoryjne, 31
klastrowanie, 26
klasyfikacja, 26
konsolidacja, 20, 25
kwerendowanie, 23, 29
liczbowe, 31
mapowanie, 25
multimedialne, 31
nieustrukturyzowane, 31
normalizacja, 21, 25, 29
oczyszczanie, 20, 21, 29
odzyskiwanie, 24
operacyjne, 32
postprocessing, 29
pozyskiwanie, 23
preprocessing, 22, 25, 29
przetworzone, 32
scalanie, 25
surowe, 32
tekstowe, 31, 46

dane
transformacja, 24, 30
ujednolicanie, 22
ustrukturyzowanych, 31
uzupełnianie następcze, 20
uzupełnianie wsteczne, 20
walidacja, 20, 353
wydobywanie, 26, 30
wzbogacanie, 22, 29
zabrudzone, 32
zagregowane, 32
zbieranie, 23
zeskrobywanie, 24
zwalidowane, 32
żniwa, 23
data
aggregation, 19, 22
classification, 26
cleansing, 20
clustering, 26
collecting, 19
consolidation, 20
enrichment, 19, 22
exchange/interchange, 215
exploration, 24
extraction, 29
harvesting, 19
integration, 19, 25
lakes, 40
mapping, 25
mart, 40
marts, 40
merging, 25
mining, 26, 30
munging, 24, 25
normalization, 21
querying, 19, 23
recovery, 19, 24
reshaping, 19, 24, 25
retrieval, 19, 23
schema, 38
scraping, 19, 24
transformation, 24
validation, 20
warehouses, 40
wrangling, 19, 24, 25
Database Management System, 23
databases, 39
decision tree, 26
decode, 160
decomposable characters, 126
decymalny, 80
decyzyjne drzewo, 26
deduplikacja, 20
definicja krotki, 37
deklaracja, 255, 355, 360, 373
globalna, 373
podstawowa, 255
XML, 255
dekodowanie, 148, 154, 156
dekomponowalne, 126
delete, 23
delimiter, 225, 227, 228
Delta Lake, 41
deserializacja, 216
deskryptywny, 217
diakrytyczny, 123, 130
dialekt, 226–228, 283, 339
DictReader, 224–226
dicttoxml, 266–268
DictWriter, 225–227
dimensionality reduction, 26
dimensions, 32
dominanta, 20, 24
DOS, 107, 133
double quadruple word, octuple word, 37
double word, dword, 37
doublequote, 227
drzewiasta, 26
drzewo
decyzyjne, 26
derywacyjne, 28
sinusoidalne, 51
struktura drzewiasta, 26
zależności, 28
zależnościowe,28
DTD, 255–258
dump, 235, 248
dumps, 235
DVD, 50
dwójkowy, 33, 34, 84, 90, 95, 96
dwukropkowa, 294
dwusłowo, 37
dyrektywa, 160, 240, 245
dyskretyzacja, 22
dziesiętny, 33, 34, 83, 95, 96, 97, 140
dźwięk, 45, 49, 51

E

EBNF, 275, 276, 282, 283
ECMAScript, 168, 176, 179–182, 193, 195
eksploracja, 24, 30
danych, 19
ekstrakcja, 19, 26, 29, 30
ekstrakcji danych, 19

ekwiwalencja, 125
  kanoniczna, 125
element, 38, 68, 79, 191, 208, 220, 251, 252, 257–259, 263, 264, 267, 269, 278, 292, 297, 299, 301, 312, 315, 319, 332, 334, 341, 344, 350, 356, 357, 360–383
  główny, 220
  zastępczy, 377
ELT, 30
encja, 38, 252, 253, 257
  ogólna, 257
  parametryczna, 257
  predefiniowana, 257
  tekstowa, 257, 258
  wewnętrzna, 257
  zewnętrzna, 258
encode, 69, 70, 160
encoding, 70, 97, 155
end, 127, 207, 210
endianness, 36, 142, 144, 153–155
endpoint, 41
ensure_ascii, 235
entity, 38
escape, 102, 134, 201, 202, 208, 240
escapechar, 227
etka, 100, 101
ETL, 30
ewaluator, 286, 291
expand, 208
eXtensible Markup Language, 251

F

fakt, 32
field, 37
fieldnames, 225, 226
find, 264
findall, 203, 204
finditer, 204, 209, 211
Flash, 347
flat data files, 41
flat files, 41
format danych, 215
frequency analysis, 27
fromstring, 263
fullmatch, 202, 203, 211
funkcje
  agregujące, 311
  escape, 201
  konwertujące, 314
  napisowe, 308
  string, 314
  strumieniowe, 67
  użytkownika, 320

G

Gellish, 215
generowanie chmur wyrazów, 27
geoprzestrzenne, 31
get, 178, 264
glif, 123
glob, 63, 64
głębia bitowa, 50–53
Golang, 173, 176, 179, 181, 182, 195
Google BigQuery, 40
Google Cloud Storage, 41, 42
Google Drive, 42
Google Sheets, 42
graf, 28, 40, 220
grafika
  pikselowa, 53
  rastrowa, 53
gramatyka, 273–278, 281, 282
  formalna, 273
  parsująca, 275, 278, 280
  w notacji EBNF, 282
  w standardzie EBNF, 275
granica, 22, 113
grawisowy, 101, 104
greedy quantifiers, 175
gromadzenie, 23
  danych, 19
group, 205, 206, 207
groupdict, 206
groups, 202, 206
grubokońcówkowość, 143
grupowanie, 26, 177
guess_type, 68, 69
GZIP, 348

H

Hadoop Distributed File System, 41
halfword, 37
hapax legomena, 20, 21
harvesting, 23
HDF5, 347
heksadecymalny, 80
hex, 95
hex digit, 33
hex editor, 46
hiperonim, 45
HTML, 24, 57, 100, 167, 217, 260, 319, 347
HTTP, 42
human-readable, 48
hurtownia danych, 40

I

ids, 267, 268
IE Tab, 258
imputacja, 20
indent, 235, 266
IndexError, 207
InfluxDB, 40
informatyka kwantowa, 38
insights, 26
instrukcja przetwarzania, 255, 256
int, 95
integracja, 25
  danych, 19
  schematu, 25
Internet Explorer, 258
Internet of Things, 42
internet rzeczy, 42
interpreter, 168, 318
IoT Devices, 42
IP, 339
ISO 8859, 98, 105–107, 110, 111, 136–139
item_func, 267–269
iter, 263, 285, 286

J

Java, 100, 199, 232, 318
JavaScript Object Notation, 230
jezioro danych, 40
język, 29, 217–219, 251
  deskryptywny, 217
  opisowy, 217
  proceduralny, 217
  przedstawieniowy, 217
  reprezentacyjny, 217
  semantyczny, 217
  znacznikowy, 100, 217
JPG, 347
JSON, 27, 31, 32, 40, 42, 93, 215, 217–219, 224, 230–235, 246, 276, 278, 280, 282, 287, 289, 294, 323–325, 334, 340, 342, 346, 348, 350, 351, 353
JSON Path, 291–294, 297, 298, 301
JSON Pointer, 287–294, 297
jsonpointer, 288
jsonschema, 351, 352

K

kanonicznie ekwiwalentne, 125
kapitalikowej, 124
karetka, 101
kasztowość, 80, 198, 253
kategoryjne dane, 31
kilobajt, 35
klastrowanie, 26
klasyfikacja danych, 26, 127
klingoński, 109
klucz, 378
kod odpowiedzi HTTP, 42
kod sterujący, 127, 129, 158
kodowanie, 97, 98, 105, 108, 109, 142, 146, 147, 153, 156
  8-bitowe, 146
  ASCII, 98, 99
  bezpośrednie, 146, 147
  CP-850, 107
  CP-852, 98, 107
  ISO 8859-2, 98, 105
  Latin-1, 106
  Latin-2, 106
  rozdzielcze, 146, 147
  rozpoznawanie, 156
  UCS-2, 98
  UTF-32, 98
  UTF-8, 98
  Windows-1250, 107
  Windows-1252, 107
kolizja nazw, 351
kombinacja, 155
komentarz, 200, 232, 237, 256, 313, 332
konsolidacja danych, 20, 25
konstruktor, 248
konsumowalność, 180
konwersja ułamków, 91
korzeń, 220, 263, 268, 298, 333, 350
kotwica, 244
krotka, 37, 68
kubit, 38
kwantyfikatory, 174
  leniwe, 175
  własnościowe, 175
  zachłanne, 175
kwantyzacja, 49, 50
kwerenda, 39
kwerendowanie danych, 19, 23, 29

L

lark, 283
LaTeX, 101, 217
Latin 1, 107
Latin 2, 107
lemat, 21
lematyzacja, 21

LibreOffice Calc, 42
ligatura, 125
lineterminator, 227, 228
liniowa, 26
listowanie plików, 63
litera diakrytyzowana, 124
literał, 95, 171, 172
Little Endian, 144–146, 155
load, 234, 247
loads, 234
logarytmizacja, 21
lukier składniowy, 64
lxml, 264, 315, 318

Ł

łączące znaki diakrytyczne, 129
łączenie danych, 25

M

mapowanie danych, 25
mapping, 25
markup, 217
match, 203
matryca podstawiania, 189
mebibajt, 36
megabajt, 35, 36
merging, 25
MessagePack, 215
metadane, 31, 32, 46, 50, 53, 73, 218,
metajęzyk, 219, 220
metasekwencja, 169, 180
metaznak, 169, 176, 180, 190, 191, 198
Microsoft Azure Blob Storage, 42
Microsoft Azure Data Lake Storage, 41
Microsoft Azure Synapse, 40
Microsoft Excel, 42, 347, 382
Microsoft OneDrive, 42
Microsoft Power Automate, 42
Microsoft SharePoint, 42
Microsoft SQL Server, 39
Microsoft Word, 130
migracja, 30
mimetypes, 68
minidom, 265
młodszych bitach, 33
mode, 68, 69, 72
mode modifiers, 197
modelowanie tematyczne, 27
MongoDB, 40
MP4, 347
MPEG, 347
Multimedialne dane, 31
MySQL, 39

N

name, 68
n-gram, 27
named entity recognition, NER, 27
napięcie elektryczne, 49
napis pusty, 71, 72, 167, 179, 276
NDJSON, 215
negative lookahead, 194
negative lookbehind, 194
Neo4j, 40
neural networks, 26
niski surogat, 147
noncharacters, 129
normal forms, 125
normalizacja, 21, 25, 29, 125
  danych, 19
normalization forms, 125
notacja
  EBNF, 282
  naukowa, 93
  nawiasowa, 292
Notepadd++, 144
null, 38, 145, 334
numeric data, 31
nybble, 33
nybl, 33

O

object_hook, 234
object_pairs_hook, 234
OCR, 20
oct, 95, 96
oczyszczanie danych, 19
odwołania, 176–179, 196, 204, 327, 331, 374
  bezwzględne, 178
  dynamiczne, 331
  rekurencyjne, 327
  względne, 178
odzyskiwanie danych, 19, 24
ogólna postać liczby, 77
ograniczenia
  danych binarnych osadzonych w tekście, 346
  liczb, 341
  napisów, 338
  obiektów, 342
  tablic, 341

oktalny, 80
open, 63, 64
OpenOffice Calc, 42
operacyjne dane, 32, 133
operand, 319
operator
  arytmetyczny, 308
  logiczny, 306
  porównania, 319
  warunków, 348
Oracle Database, 39
ord, 140
outliers, 21
oznakowany tekst, 217

Ó

ósemkowy, 33, 34, 95, 96

P

pangram, 157, 158
para klucz-wartość, 237
para uporządkowana, 37
Parquet, 215
parse, 263
parse tree, 28
parse_constant, 234
parse_float, 234
parse_int, 234
parser, 57, 227, 232, 237, 243, 248, 249, 256, 260, 261, 323, 371
parsing, 27
parsing data, 19
parsowanie, 21, 24, 25, 27, 28, 41, 227, 232, 263, 273, 278, 281, 323, 346
PascalCase, 21
patterns, 26
PDF, 129, 217
PHP, 199, 347
placeholder, 299
plain text, 31, 46, 243
plik
  binarny, 45
  dźwiękowy, 45
  graficzny, 45
  płaski, 41
  tekstowy, 45
  wideo, 45
PNG, 347
podstawianie, 176, 177, 208
pole, 38, 39, 51, 129, 183, 190
positive lookbehind, 194
POSIX, 172, 173, 182
possessive quantifiers, 175
postać
  w pełni skomponowana, 125
  w pełni zdekomponowana, 125
  znormalizowana, 125
PostgreSQL, 39
postprocessing, 29
potomek, 220, 263
pozycja kodowa, 110
pozyskiwanie, 23
  danych, 19
półbajt, 33–35
półsłowo, 37
predefiniowane atrybuty, 261
predicates, 301
predykat, 301
prefiks, 140, 149–154, 260
preprocessing, 22, 25, 29
processing instructions, 256
Protobuf, 215, 347
przechwytywanie, 176, 177
przenoszalny, 216
przestrzeń nazw, 259, 320, 325, 355, 356
  XML, 356
przetworzone dane, 32, 143
pseudografika, 130
punkt kodowy, 110, 125
punkt końcowy, 41
Python, 63, 69, 91, 95, 100, 140, 160–163, 168, 176, 178, 179, 182, 193, 195, 199, 223, 224, 234, 247–249, 263, 268, 280, 318
PyYAML, 244, 247

Q

quad word, 37
quadbit, 33
quadruple word, 37
quantum computing, 38
queries, 39
QUOTE_ALL, 227
QUOTE_MINIMAL, 227
QUOTE_NONE, 227
QUOTE_NONNUMERIC, 227
quotechar, 225, 227
quoting, 227
qword, 37

R

ramka, 50, 152, 156
  binarna, 151, 156
RDF, 215, 260
rdzeniowanie, 20, 21
read, 67
reader, 224, 225, 227
readinto, 68
readline, 67
readlines, 67
REBOL, 215
reconstructable, 216
Redis, 40
redukcja wymiarów, 26
regexes, 167
regexps, 167
regresja, 26
  liniowa, 26
  prosta, 26
  wielomianowa, 26
regression, 26
regular expressions, 167, 201
reguła przepisywania, 273
rekord, 37
reprezentery, 248
restkey, 225
return_bytes, 267
RFC 1123, 339
RFC 14212, 58
RFC 20453, 58
RFC 21524, 58
RFC 2673, 339
RFC 35015, 58
RFC 3986, 339
RFC 3987, 340
RFC 4122, 339
RFC 4648, 58
RFC 4648, 58
RFC 48808, 58
RFC 5322, 339
RFC 5890, 339
RFC 6531, 339
RFC 6570, 340
RFC 6901, 287
RFC 6901, 340
RGB, 55
rodzeństwo, 220, 298
rodzic, 220, 329
rozdzielcze, 147
rozpoznawanie, 26, 27, 156
rozszerzalność, 251
rozwiązywacze, 248
rozwiązywanie konfliktów danych, 20
rozwinięcie liczby, 78

S

safe_load, 247
sample rate, 50
sample, 49
sampling, 49
Saxon, 318
scalanie danych, 25
schemat danych, 38
schemat pusty, 354
scraping sources, 42
search, 202, 203, 205, 207, 208, 209, 211
seek, 68
semigrafika, 130
semi-structured data, 31
sentyment, analiza, 22, 27
separator, 128, 135, 183, 190–192, 222, 227, 235, 240
  kolumn, 227
  wierszy, 227
separators, 235
serializacja, 216
server side API, 41
sieci neuronowe, 26
silnik, 168
skipinitialspace, 227
skipkeys, 235
składacz, 263, 374
składanie, 199
składnica danych, 40
słowo, 36
  długość, 36
  maszynowe, 36
  ośmiokrotne, 37
  poczwórne, 37
  podwójne, 37
  stopujące, 20
snake_case, 21
Sniffer, 228
Snowflake, 40
sort_keys, 235
spacja, 100, 102, 126, 134, 135, 137, 173, 235, 238, 240, 243
span, 207, 209, 210, 211
split, 208
spreadsheet, 42
SQL, 23, 29, 318, 319, 347

standard
ISO 10646, 110
ISO 8601, 241, 339
RFC 1123, 339
RFC 14212, 58
RFC 20453, 58
RFC 21524, 58
RFC 2673, 339
RFC 35015, 58
RFC 3986, 339
RFC 3987, 340
RFC 4122, 339
RFC 4648, 58
RFC 48808, 58
RFC 5322, 339
RFC 5890, 339
RFC 6531, 339
RFC 6570, 340
RFC 6901, 287, 340
starszy bit, 33
start, 207
startowy symbol, 283, 286
stat, 73
stereo, 49, 51
stop word, 20
stopujące słowo, 20
storable, 216
strict, 70, 227, 371
string, 208
strona kodowa, 97
structured data, 31
struktura drzewiasta, 26
strumień, 64, 68
sub, 202, 208, 210
subn, 202, 208, 210
subtraktywny, 83
surogat, 147, 148
surowe dane, 25, 40
survival analysis, 26
symbol
nieterminalny, 275
startowy, 283
terminalny, 273
zastępczy, 299
syntactic sugar, 64
System.Text.RegularExpressions, 168
system zarządzania bazą danych, 39
system kodowania znaków, 97
system zapisu liczb
addytywny, 82
binarny, 84
dwójkowy, 84
niepozycyjny, 82
pozycyjny, 82
substraktywny, 82
szesnastkowy, 33, 34, 90
szereg czasowy, 40

Ś

ścieżka, 63, 68, 71, 263, 267, 291, 297–299, 308, 312
środowisko programistyczne, 382

T

tablica, 51, 53–56, 97–99, 106–113, 122–124, 149, 158, 160, 230–232, 287, 288, 333–338, 341–345, 348
kodowa, 97–99, 101–109, 130, 131, 158
tag, 245, 248, 264
TAR, 346
tebibajt, 36
tekst
bogaty, 45
czysty, 46, 48, 243
Telegazeta, 130
tell, 68
terabajt, 35, 36
text, 264
text mining, 19, 27
text summarizing, 27
time series analysis, 26
token, 21, 358
tokenizacja, 21
TOML, 215
ton, 51, 52
transakcje, 39
transferable, 216
transformacji danych, 19, 24, 30
Transformer, 286
TSV, 218, 222, 223, 232, 233
tuple, 37
TXT, 222, 223
typ danych, 38, 223, 230, 231, 238, 241

U

ujednolicanie kodowania, 21
ukośnik, 101, 103, 104, 169, 172, 240, 252, 261, 262, 287, 288, 298, 378
Unikod, 98, 108–126, 129–132, 230, 240
unstructured data, 31
update, 23
URI, 260, 314, 325, 329, 340, 357, 359
URL, 57, 58, 245, 325, 327, 354, 359

urządzenia IoT, 42
usługi chmurowe, 42
usuwanie, 23
  białych znaków, 21
  interpunkcji, 21
  rekordu, 20
UTF-16, 146
UTF-32, 145
UTF-8, 149, 151, 152, 153
uzupełnianie wsteczne, 20

V

Visual Studio, 382

W

walidacja, 20
walidator, 325, 371
wartości
  atomowe, 37, 38
  stałe, 344
  unikatowe, 378
web scraping, 24
węzeł, 248
whitespace characters, 126
wideo, 31, 47, 49, 347
wielomianowa, 26
wierzchołek, 220
właściwość, 264
word cloud generation, 27
word embeddings, 27
word wrapping, 127
write, 67, 68, 224–226, 265
writeheader, 225, 226
writelines, 68
writer, 224–227
wstępne przetwarzanie, 29
WWW Consortium, 251
wydobywanie danych, 19
wykrywanie anomalii, 26
wymiana danych, 215, 216
wymiary, 32
wyrażenia
  FLWOR, 319
  regularne, 167–169, 178, 189, 193, 197, 201–208, 281
  warunkowe, 195, 196, 348
wysoki surogat, 147
wzbogacanie, 22, 29
  danych, 19

X

XHTML, 260, 347
XML, 31, 32, 42, 70, 93, 100, 167, 215, 217–220, 224, 232, 246, 251–269, 273, 277–282, 297, 318, 346, 354–360, 374, 381–386
XML Schema, 219, 260, 320, 354–358, 381, 385
xml_declaration, 267
xmlschema,
XPath, 264, 297–299, 306, 308, 311, 315, 318, 319, 378
XQuery, 318, 319, 320
XSLT, 217, 256, 260

Y

YAML, 93, 215, 217, 219, 224, 237–249, 253, 257, 282, 346

Z

zabrudzone dane, 30
zagregowane dane, 22, 32
zapytania, 39
zawijanie wierszy, 127
zażółć gęślą jaźń, 157
zbieranie danych, 19, 23
zeskrobywanie danych, 19, 24
ZIP, 348
zmiennoprzecinkowa liczba, 340, 358
znacznik, 154, 155, 217, 218, 232, 233, 237, 243, 244, 251, 252, 255, 273
znak, 100
  biały sensu largo, 126
  biały sensu stricto, 126
  biały, 100
  BOM U+FEFF, 129
  końca linijki, 127
  niedrukowalny, 127
  niewidoczny, 127
  prekomponowany, 126
  sterujący, 100, 126
  Unicode, 182
znakowanie, 217

Ź

źródła zeskrobywalne, 42

Ż

żniwa danych, 23

Dane: załaduj, przetwarzaj, analizuj

Ekstrakcja danych jest procesem, w którym informacje pozyskuje się z różnych źródeł — zwykle po to, by następnie poddać je dalszej transformacji i analizie. Umiejętność pozyskiwania danych, scalania, filtrowania i obrabiania ich na rozmaite sposoby przydaje się nie tylko zawodowym analitykom. Zdolność poruszania się po świecie danych jest wysoce pożądana również u osób pracujących w działach IT i na stanowiskach menadżerskich. Kto ma dane, ten ma wiedzę i zyskuje przewagę nad konkurencją!

Jeśli chcesz zgłębić teorię ekstrakcji danych i zdobyć praktyczne umiejętności pozwalające operować nimi w Pythonie, ten podręcznik powinien być dla Ciebie pozycją obowiązkową.

Dzięki książce między innymi:

- Opanujesz podstawowe pojęcia, których znajomość jest niezbędna podczas działań na zbiorach danych
- Zrozumiesz specyfikę plików binarnych i tekstowych
- Dowiesz się, na czym polega kodowanie tekstu
- Poznasz zagadnienia wyrażeń regularnych
- Zorientujesz się, jakie formaty wymiany danych są dostępne w Pythonie
- Nauczysz się przeszukiwać dokumenty znacznikowe
- Zapoznasz się ze schematami formatów wymiany danych

Piotr Rybka, doktor nauk humanistycznych, polonista, językoznawca, informatyk. Prowadził zajęcia na Uniwersytecie Śląskim i w Akademii Górniczo-Hutniczej. Pracował w Instytucie Języka Polskiego PAN. Autor książek i artykułów. Zainteresowanie fonetyką doprowadziło go do programowania w C# i Pythonie, których używał do analizy akustycznej, automatycznej transkrypcji i generowania tekstów języka naturalnego. W czasie wolnym kolekcjonuje dobrą muzykę na płytach CD.

Wydawnictwo Naukowe Helion

helion.pl

HELION S.A.
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

KOD KORZYŚCI
Sięgnij po więcej!

ISBN 978-83-289-2169-6

9 788328 921696

Cena: 99,00 zł