Politechnika Świętokrzyska
Wydział Zarządzania i Modelowania Komputerowego

Kierunek studiów:
Inżynieria danych

Aleksandra Sikora

Materiały dydaktyczne do przedmiotu

BAZY DANYCH TYPU BIG DATA

opracowane w ramach realizacji Projektu
„Dostosowanie kształcenia w Politechnice Świętokrzyskiej do potrzeb współczesnej gospodarki”
FERS.01.05-IP.08-0234/23

Kielce, 2025

Spis treści

1. Wstęp

Niniejsze materiały dydaktyczne stanowią uzupełnienie do przedmiotu „Bazy danych typu Big Data", realizowanego na VI semestrze kierunku „Inżynieria danych". Mogą być również przydatne studentom innych kierunków – zarówno przy realizacji prac dyplomowych, jak i projektów obejmujących szerokie spektrum tematów związanych z danymi.

Skrypt ma charakter pomocniczy: nie zastępuje wykładów, laboratoriów ani literatury przedmiotu. Podstawowe pojęcia, definicje i ujęcia teoretyczne należy czerpać przede wszystkim z zajęć oraz z pozycji wskazanych w rozdziale „Literatura". Założono, że czytelnik posiada przygotowanie z zakresu baz danych oraz języka programowania Python.

W opracowaniu przyjęto następujące konwencje:
- podano alternatywne nazwy w nawiasach (lub krótkie wyjaśnienia), aby ułatwić kojarzenie terminologii z różnych źródeł,
- terminy, których wyjaśnienie zaburza ciąg zdania zostały oznaczone *, a ich wyjaśnienie zamieszczono w przedostatnim rozdziale,
- sformułowano zalecenia minimalne, które spełniają wymagania dobrego projektu, oraz wskazano możliwe rozszerzenia,
- konsekwentnie stosowano słowo „projekt" w szerokim znaczeniu: od zadania realizowanego w laboratorium po całościowe rozwiązanie problemu.

Materiały dydaktyczne opracowano na podstawie dokumentacji dostępnych na wrzesień 2025. Ze względu na możliwe zmiany w dokumentacji, API (Application Programming Interface), cennikach usług chmurowych, licencjach oraz strukturze i dostępności zbiorów, część informacji może ulec zmianie, dlatego przed użyciem w projekcie zaleca się weryfikację.

2. Repozytoria i katalogi danych

W rozdziale scharakteryzowano wybrane źródła danych. Przedstawiono praktyczne wskazówki jak je dobierać do projektów czy prac dyplomowych. Wyszczególniono, w jakich sytuacjach które repozytorium daje największą wartość.
Przedstawiono przede wszystkim repozytoria i katalogi danych wykorzystywane w dydaktyce oraz w projektach uczenia maszynowego. Zawarte tam zbiory mają często małą lub średnią skalę i służą jako materiał do prototypowania, benchmarków oraz ćwiczeń z replikowalnością (metadane, wersjonowanie, DOI*). Gdy potrzebna jest analiza danych w rzeczywistej skali „big data" (formaty kolumnowe, partycje,

praca in-place bez pobierania), odwołujemy się do rozdziału 3 dotyczącego platform chmurowych z danymi publicznymi.

2.1. Kaggle Dataset

Kaggle (Kaggle, b.d.-a) to społecznościowa platforma dla analizy i inżynierii danych. Łączy publiczne repozytorium zbiorów danych Kaggle Datasets (Kaggle, b.d.-b), uruchamialne w chmurze Kaggle Notebooks (Kaggle, b.d.-c), interfejs Kaggle API (Kaggle, b.d.-d) do automatyzacji pobierania i zarządzania zasobami oraz ekosystem konkursów modelowania. W zastosowaniach profesjonalnych platforma służy do publikacji i wersjonowania danych, szybkiego prototypowania i odtwarzalnych eksperymentów, benchmarkingu (porównawcza ocena metod lub systemów na wspólnym zbiorze danych i metrykach) modeli w ramach konkursów oraz współpracy zespołowej nad wspólnymi zasobami danych i kodu. Zapewnia wyszukiwanie po tagach i formatach, bogate metadane i historię wersji, a także mechanizmy udostępniania wyników i integracji z konfiguracjami CI (Continuous Integration, ciągła integracja - automatyczne budowanie, testy i walidacje uruchamiane przy każdej zmianie w repozytorium kodu). Katalogi Kaggle ułatwiają szybkie prototypowanie i porównania oraz automatyzację pobierania danych przez API. Przykład: w studium przypadku (Wilk-Jakubowski, Sikora, & Maciejski, 2025) wykorzystano publiczny zbiór adresów URL z Kaggle do przeprowadzenia badań.

2.2. Mendeley Data

Mendeley Data (Mendeley. b.d.) to repozytorium danych badawczych Elseviera (globalne wydawnictwo naukowe i dostawca narzędzi informacyjnych, wydawca i operator m.in. ScienceDirect i Scopus) służące do deponowania, publikacji i cytowania zbiorów danych. Zapewnia nadawanie identyfikatorów DOI dla datasetów, wersjonowanie i bogate metadane, w tym powiązania z ORCID (Open Researcher and Contributor ID, unikalny identyfikator autora) oraz opisami dyscyplin i słów kluczowych. Udostępnia opcje embarga (czasowe wstrzymanie publicznego dostępu do datasetu do wskazanej daty lub wydarzenia), wybór licencji (np. CC BY, CC0) i generowanie cytowań, a także integrację z publikacjami naukowymi, co ułatwia weryfikowalność i ponowne wykorzystanie danych. W praktyce profesjonalnej pełni rolę stabilnego, cytowalnego źródła danych do badań, audytowalnych eksperymentów i współpracy zespołowej nad aktualizowanymi wersjami zasobów.

2.3. UCI Machine Learning Repository

UCI Machine Learning Repository (Kelly, Longjohn, & Nottingham, b.d.) to repozytorium zbiorów danych do uczenia maszynowego, utrzymywane przez University of California, Irvine. Udostępnia zbiory głównie tabelaryczne z opisami atrybutów, źródłem danych, zadaniem (klasyfikacja, regresja, klasteryzacja) oraz odniesieniami do publikacji, co ułatwia replikację i cytowanie. Pliki są zazwyczaj w formatach tekstowych, z towarzyszącą dokumentacją i metadanymi, a warunki licencyjne zależą od właściciela konkretnego zbioru. Zalecana jest weryfikacja licencji na stronie danego datasetu. Repozytorium bywa wykorzystywane do budowy benchmarków, trenowania modeli uczenia maszynowego (machine learning, ML) i porównań na wspólnych metrykach. Wiele zbiorów ma niewielką skalę, dlatego w projektach Big Data służy często jako punkt odniesienia lub materiał do prototypowania przed pracą z danymi wielkoskalowymi.

2.4. OpenML

OpenML (OpenML, b.d.) (Vanschoren et al., 2014) to otwarta platforma do współdzielenia zbiorów danych, zadań i wyników eksperymentów ML z metadanymi oraz API. Platforma definiuje cztery podstawowe elementy pracy: dataset (zbiór danych), task (zadanie z ustaloną metryką i protokołem ewaluacji), flow (implementacja modelu lub potoku) oraz run (konkretne uruchomienie na danym zadaniu), co ułatwia replikowalność i porównywanie wyników w jednolitych warunkach. Projekty mogą korzystać z oficjalnych bibliotek (dla Pythona i R), które pozwalają automatyzować eksperymenty oraz uzyskiwać dostęp do zasobów przez API.

2.5. Jak dobrać repozytorium

Gdy potrzebny jest szybki start pracy, łatwy dostęp do przykładów oraz wsparcie społeczności, praktycznym wyborem jest Kaggle Dataset. Platforma udostępnia publiczny katalog zbiorów danych, uruchamialne w chmurze notatniki oraz interfejs API do automatyzacji pobierania i zarządzania danymi, co przyspiesza realizację projektów i szybkie prototypowanie. Dzięki wbudowanym metadanym i wersjonowaniu łatwo jest odtwarzać wyniki oraz dzielić się kodem ze studentami i współautorami.

Jeśli celem jest udostępnienie danych w sposób trwały i możliwy do jednoznacznego cytowania, właściwa będzie usługa Mendeley Data. Repozytorium nadaje identyfikatory DOI (DataCite), wspiera wersjonowanie, umożliwia wybór licencji oraz

ustawienie embarga, co ułatwia proces recenzji i późniejsze ponowne wykorzystanie danych w badaniach lub dydaktyce.

Do kursów wprowadzających i porównań metod na niewielkich, dobrze opisanych zbiorach przydatne jest UCI Machine Learning Repository. Zbiory mają spójne opisy atrybutów i zadań, a skala danych pozwala skupić się na zrozumieniu algorytmów oraz metryk bez rozpraszania się kwestiami infrastrukturalnymi.

Gdy potrzebne jest rejestrowanie całego cyklu eksperymentów wraz z definicjami zadań, przepływów i uruchomień, właściwym wyborem jest OpenML. Platforma wprowadza pojęcia dataset, task, flow i run, a biblioteki klienckie dla Pythona i R ułatwiają replikowalność, benchmarking oraz automatyzację większych serii eksperymentów.

Kryteria doboru wiarygodnych repozytoriów zostały szerzej opisane w pozycji (Pawłowska & Wachowicz, 2020).
W praktyce badawczej zbiory z wymienionych repozytoriów są powszechnie wykorzystywane, co można np. zaobserwować w przeglądzie literatury z lat 2017-2024 (Wilk-Jakubowski, Pawlik, et al., 2025) pokazującym liczne prace oparte na danych z m. in. UCI, Kaggle Dataset i Mendeley Data.

3. Platformy chmurowe z danymi publicznymi

3.1. BigQuery Public Datasets

BigQuery Public Datasets (Google Cloud, 2025) to utrzymywany przez Google katalog ogólnodostępnych zbiorów danych w projektach „bigquery-public-data”, które można analizować bezpośrednio w BigQuery (in-place), używając standardowego SQL (Structured Query Language). Zbiory obejmują różne domeny (administracyjne, ekonomiczne, naukowe, geograficzne, cyberbezpieczeństwo, dane z serwisów społecznościowych), mają jawnie zdefiniowane schematy tabel, często są partycjonowane/klastrowane i regularnie aktualizowane przez kuratorów lub właścicieli danych. Dostęp jest możliwy przez Konsolę Cloud, narzędzie wiersza poleceń bq oraz API, a także przez popularne biblioteki w Pythonie (google-cloud-bigquery, pandas-gbq) i R (bigrquery). Typowym wzorcem pracy jest odwołanie do tabel w przestrzeni bigquery-public-data i budowa własnych zapytań, widoków oraz modeli (np. BigQuery ML) bez kopiowania danych. Każdy zbiór posiada opis i metadane (schemat, pochodzenie, zakres czasowy, częstotliwość odświeżania), a prawa i warunki wykorzystania są określane przez dostawcę danego datasetu.

3.2. AWS Registry of Open Data

AWS Registry of Open Data (RODA) (Amazon Web Services, b.d.-a) to publiczny katalog otwartych zbiorów danych utrzymywanych w chmurze Amazon Web Services. Zamiast jednego centralnego repozytorium, rejestr gromadzi opisy datasetów publikowanych głównie w Amazon S3 przez instytucje i zespoły badawcze (często w ramach programu Open Data Sponsorship). Każdy wpis zawiera metadane ułatwiające pracę dydaktyczną i badawczą: zakres tematyczny, strukturę i format plików (np. Parquet*, CSV), region i położenie zasobów (URI S3), częstotliwość aktualizacji, licencję oraz zasady cytowania. Dane są tylko do odczytu i analizuje się je „in-place" narzędziami AWS bez konieczności kopiowania do własnej infrastruktury. Rejestr obejmuje różnorodne domeny, m.in. meteorologię i klimat, obserwacje Ziemi (obrazy satelitarne), geodezję, zdrowie publiczne, dane ekonomiczne i ruch sieciowy. Wpisy często zawierają wzorcowe zapytania lub notatniki startowe. W kontekście zajęć akademickich kluczowe jest korzystanie z metadanych rejestru do wyboru odpowiedniego formatu i regionu danych, jak również weryfikacja licencji i wymogów cytowania wskazanych przez dostawcę konkretnego zbioru.

3.3. Azure Open Datasets

Azure Open Datasets (Microsoft, 2025; Microsoft, b.d.) to kuratorowany katalog publicznie dostępnych zbiorów danych hostowanych w ekosystemie Microsoft Azure, przygotowany do analizy „in-place" w usługach analitycznych i uczenia maszynowego (ML, Machine Learning). Zbiory są udostępniane głównie w Azure Blob Storage lub Azure Data Lake (ADLS) (często w formatach kolumnowych, np. Parquet), z opisanymi schematami, zakresem czasowym i źródłem pochodzenia. Typowe domeny obejmują m.in. transport, dane demograficzne i administracyjne, zdrowie publiczne, pogodę i klimat oraz obserwacje Ziemi. Dostęp i praca odbywają się przez narzędzia takie jak Azure Synapse Analytics, Azure Databricks (Apache Spark), Azure Machine Learning (notatniki, pipeline'y) oraz interfejsy SDK/REST API. Wiele pozycji ma gotowe przykłady zapytań lub przykładowe knotebooki startowe ułatwiające zajęcia laboratoryjne. Warunki wykorzystania i zasady cytowania są określane na poziomie konkretnego zbioru, należy je sprawdzić w metadanych przed włączeniem datasetu do ćwiczeń, projektów czy prac dyplomowych.

3.4. Podsumowanie kosztów

W kontekście zajęć dydaktycznych możliwe jest korzystanie z publicznych zbiorów danych bez opłat za samo składowanie, przy rozliczaniu głównie zapytań i ewentualnego transferu.
W Google BigQuery dostępny jest tryb Sandbox działający bez karty i w ramach darmowych limitów, odpowiedni do ćwiczeń laboratoryjnych (przykłady w kolejnych rozdziałach) (Google Cloud, b.d.-a; Google Cloud, b.d.-b). W modelu rozliczeń on-demand BigQuery nalicza minimum 10 MiB* na zapytanie oraz dodatkowe minimum 10 MiB na każdą tabelę użytą w zapytaniu. Zbyt niski --maximum_bytes_billed (np. 1 MiB) zakończy zapytanie błędem, nawet gdy szacowane przetwarzanie jest mniejsze.
Na AWS publiczne zbiory danych można analizować in-place narzędziem Amazon Athena (Amazon Web Services, b.d.-b), płacąc tylko za zapytania, bez pobierania danych do własnej infrastruktury. Athena rozlicza zapytania za bajty zeskanowane, z minimum 10 MB na zapytanie, zaokrąglając do 1 MB (Amazon Web Services, b.d.-b).
W Azure Open Datasets składowanie jest zapewniane przez dostawcę, natomiast koszty dotyczą obliczeń i egressu (dane opuszczające sieć lub chmurę).

Programy edukacyjne ułatwiają bezkosztowe początki: BigQuery Sandbox dla ćwiczeń, Azure for Students z przyznawanymi kredytami bez karty oraz AWS Educate z bezpłatnymi laboratoriami i kredytami dla studentów i wykładowców.

4. Porównanie repozytoriów danych oraz platform chmurowych z danymi publicznymi

„Repozytoria danych Big Data” to miejsca, w których autorzy udostępniają zbiory danych (zwykle do pobrania jako pliki) wraz z metadanymi i licencją. „Platformy chmurowe z danymi publicznymi” to usługi chmurowe, które oprócz katalogu danych oferują gotową infrastrukturę obliczeniową do analizy tych danych in-place (bez pobierania), zwykle w modelu płatności za zapytania i transfer.

Tabela 1. Porównanie repozytoriów danych i platform chmurowych z danymi publicznymi

Aspekt	**Repozytoria danych**	**Platformy chmurowe z danymi publicznymi**
Model dostępu	Pobieranie plików (CSV, Parquet, ZIP) na własną infrastrukturę	Analiza in-place w chmurze, dane często już w formatach kolumnowych i partycjonowane
Zakres funkcji	Hosting plików, wersjonowanie, metadane, licencje, DOI	Katalog + gotowe klastry/usługi (SQL, notatniki, serwisy ML), role/ACL, logowanie kosztów
Koszt	Zwykle bezpłatny dostęp do plików; koszty po stronie użytkownika (przechowywanie/obliczenia lokalne)	Płaci się głównie za zapytania, składowanie w chmurze i ewentualny egress
Skala i wydajność	Ograniczona przez lokalne zasoby; trzeba samodzielnie dbać o ETL i wydajność	Skalowalność i optymalizacje po stronie dostawcy (kolokacja danych i zasobów obliczeniowych)
Typowe zastosowania	Nauczanie, prototypy, benchmarki, replikowalność z DOI; małe/średnie zbiory	Analizy na dużą skalę, szybkie zapytania ad-hoc, integracje z usługami chmurowymi
Próg wejścia	Niski: pobierz i pracuj w znanym narzędziu	Wymagana podstawowa znajomość usług chmurowych i kontroli

5. Otwarte dane administracji publicznej

W tym rozdziale skoncentrowano się na otwartych danych administracji publicznej - od portali krajowych po serwisy Unii Europejskiej - z naciskiem na praktyczne pozyskiwanie do projektów. W odróżnieniu od rozdziału 2 (repozytoria i katalogi danych) i rozdziału 3 (platformy chmurowe z analizą in-place) położono tu większy akcent na specyfikę administracyjnych zbiorów: metadane i licencje ponownego wykorzystania, stabilne identyfikatory, definicje wskaźników i rewizje. Pokazano też dobre praktyki dokumentowania parametrów zapytań API i wersji danych, tak aby wyniki były porównywalne i replikowalne. Ostatni podrozdział opisuje, jak łączyć te źródła z narzędziami chmurowymi omówionymi wcześniej.
Skala danych w tym rozdziale waha się od małych tabel wskaźników po duże zbiory obrazowe i szeregi czasowo-przestrzenne; w przypadku pracy na pełnych kolekcjach lub analizy in-place w chmurze traktujemy je jako big data.

5.1. Wybrane portale krajowe danych publicznych

Portal Otwarte Dane RP (Ministerstwo Cyfryzacji, b.d.) to ogólnokrajowy katalog zasobów administracji publicznej, umożliwiający wyszukiwanie i pobieranie danych w formie plików (np. CSV, JSON, GeoJSON) oraz przez API. Każdy zbiór posiada metadane zgodne ze standardami (m.in. DCAT-AP*, Data Catalog Vocabulary Application Profile, profil aplikacyjny słownika katalogów danych), opis zakresu i częstotliwości aktualizacji oraz informację o licencji (np. CC BY, CC0), co ułatwia legalne ponowne wykorzystanie. Serwis oferuje filtry tematyczne i terytorialne, wersje/rewizje danych oraz dokumentację do API, dzięki czemu nadaje się do projektów studenckich i prototypów analitycznych. W projekcie podaj: pełny adres źródła (URL), użyte parametry zapytania/API, datę i godzinę pobrania oraz wersję/wydanie danych (jeśli podano).

Bank Danych Lokalnych (BDL) GUS (Główny Urząd Statystyczny, b.d.;2018) to centralny serwis statystyk publicznych dla poziomów gmina-powiat-województwo i kraju, oferujący wieloletnie szeregi wskaźników z metadanymi (definicje, jednostki, częstotliwość, rewizje). Umożliwia pobieranie danych przez interfejs WWW oraz programistycznie przez REST API (filtry po jednostkach TERYT - Krajowy Rejestr Urzędowy Podziału Terytorialnego, czasie i miarach; formaty JSON/CSV). BDL świetnie nadaje się do map i analiz porównawczych w projektach studenckich. W projekcie podaj: pełny adres źródła (URL), identyfikatory TERYT i inne parametry zapytania/API, datę i godzinę pobrania oraz wersję/wydanie danych (jeśli podano).

Geoportal Krajowy (Główny Urząd Geodezji i Kartografii, b.d.) to centralny serwis danych przestrzennych państwa, udostępniający mapy referencyjne (ortofotomapy, działki ewidencyjne, granice administracyjne), usługi przeglądania WMS (Web Map Service)/WMTS (Web Map Tile Service) i pobierania WFS (Web Feature Service) oraz odnośniki do państwowych rejestrów (m.in. PRG - Państwowy Rejestr Granic i Powierzchni Jednostek Podziałów Terytorialnych, EGiB - Ewidencja Gruntów i Budynków). W projektach studenckich sprawdza się do warstw bazowych i łączenia danych statystycznych (np. z BDL - Bank Danych Lokalnych (GUS)) po identyfikatorach TERYT (Krajowy Rejestr Urzędowy Podziału Terytorialnego Kraju). Warto zwrócić uwagę na układ współrzędnych warstw, ponieważ stosowane są różne układy, a ich niezgodność prowadzi do przesunięć i zniekształceń na mapie, oraz uważnie czytać metadane. W projekcie podaj: adres warstwy/usługi (URL), układ współrzędnych (CRS), użyte parametry zapytania oraz datę i godzinę pobrania; jeśli podano - numer wersji/wydania danych.

Generalna Dyrekcja Dróg Krajowych i Autostrad (GDDKiA) udostępnia dane o natężeniu ruchu drogowego na sieci dróg krajowych: wyniki cyklicznego Generalnego Pomiaru Ruchu (GPR) (Generalna Dyrekcja Dróg Krajowych i Autostrad, b.d., GPR 2025) oraz bieżące zestawienia ze Stałych Punktów Pomiarowych Ruchu (SCPR) (Generalna Dyrekcja Dróg Krajowych i Autostrad, b.d., SCPR). Zawierają m.in. średni dobowy ruch (SDR, ang. AADT), strukturę pojazdów, wyniki dla odcinków i punktów. Dane nadają się do analiz w GIS (np. mapy „heatmap", trendy między pomiarami) i do łączenia z innymi źródłami (BDL, wypadkowość). W projekcie podaj: adres źródła (URL), rok kampanii GPR, identyfikator odcinka/punktu, definicje wskaźników, a także datę i godzinę pobrania oraz wersję/wydanie danych (jeśli podano).

5.2. Wybrane portale Unii Europejskiej danych publicznych

European data (Publications Office of the European Union, b.d.-a) to oficjalny portal Unii Europejskiej będący punktem dostępu do otwartych danych z instytucji UE oraz z krajowych, regionalnych, lokalnych i tematycznych portali państw europejskich. Zbiera i standaryzuje metadane (DCAT-AP, (Publications Office of the European Union, b.d.-c)), umożliwia wyszukiwanie i pobieranie zbiorów, a także oferuje materiały edukacyjne (Academy, (Publications Office of the European Union, b.d.-b)) i przeglądarkę map. Zastępuje wcześniejsze EU Open Data Portal i European Data Portal, łącząc je w jeden serwis zarządzany przez Urząd Publikacji UE. W projekcie podaj: adres źródła (URL), identyfikator zbioru/kolekcji, użyte parametry zapytania oraz datę i godzinę pobrania; jeśli podano – numer wersji/wydania danych.

Eurostat (Eurostat, b.d.-c) to urząd statystyczny Unii Europejskiej, publikujący porównywalne statystyki dla państw i regionów NUTS (Nomenclature of Territorial Units for Statistics, Nomenklatura Jednostek Terytorialnych do Celów Statystycznych) w obszarach gospodarki, rynku pracy, społeczeństwa, środowiska i cyfryzacji. Dane dostępne są przez interfejs WWW (Data Browser, (Eurostat, b.d.-b)) oraz API w standardach SDMX/JSON (Eurostat, b.d.-a), z metadanymi (definicje, jednostki, rewizje) ułatwiającymi poprawne analizy i cytowanie. Dla projektów studenckich Eurostat jest dobrym źródłem szeregów czasowych i wskaźników „per capita", porównywalnych między krajami. W projekcie podaj: kod zestawu (dataset code), adres źródła (URL), parametry zapytania/API, datę i godzinę pobrania oraz wersję/wydanie danych (jeśli podano).

Europejska Agencja Środowiska (EEA, European Environment Agency) (European Environment Agency, b.d.) udostępnia zbiory danych i wskaźniki dotyczące środowiska, klimatu i użytkowania ziemi, w tym m.in. jakość powietrza (pomiarowe i modelowe serie), emisje zanieczyszczeń, pokrycie terenu (Corine Land Cover) oraz

obszary chronione. Dane są dostępne przez portal EEA Datahub, zwykle z kompletnymi metadanymi, możliwością pobrania w formatach tabelarycznych i GIS (Geographic Information System) oraz odnośnikami do usług OGC (Open Geospatial Consortium). Zbiory dobrze nadają się do analiz porównawczych między krajami/regionami i do łączenia z danymi statystycznymi (np. Eurostat). W projekcie podaj: nazwę/identyfikator zbioru, adres źródła (URL), typ usługi, parametry zapytania oraz datę i godzinę pobrania, numer wersji/wydania danych (jeśli podano).

5.3. Praca w chmurze bez „in-place compute”

Żaden z omówionych serwisów (BDL, Geoportal, GDDKiA, data.europa.eu, Eurostat, EEA) nie udostępnia analizy „in-place compute” porównywalnej z BigQuery Public Datasets. Praca więc może przebiegać w typowym, trzyetapowym schemacie (Venema, 2023):

1. Pobranie danych przez API/HTTP w środowisku chmurowym (notebook/ETL uruchomiony w chmurze; zapisanie wszystkich parametrów zapytań, wersji i czasu pobrania).
2. Składowanie w object storage (GCS/Amazon S3/Azure Blob) w formatach sprzyjających wydajności i reużywalności (np. Parquet dla tabel, COG/GeoTIFF dla rastrów; z metadanymi, sumami kontrolnymi, ewentualnie wersjonowaniem).

3. Analiza po załadowaniu do odpowiedniego silnika przy pomocy SQL w usługach typu BigQuery / Synapse / Athena lub przetwarzanie rozproszone w Spark / Databricks / notebooki.

Takie podejście upraszcza automatyzację (pipeline’y), ułatwia kontrolę kosztów (płacisz za zapytania i skanowane bajty, nie za lokalne utrzymanie klastrów) oraz wspiera replikowalność dzięki jednoznacznym metadanym i przechowywaniu danych w centralnym, wersjonowanym storage.

6. Implementacja - Budowa danych wejściowych i pierwsze zapytania w chmurze BigQuery

Zadanie:

Zrealizuj w Google Cloud Platform (GCP), w regionie EU, podstawowy przepływ pracy oparty na BigQuery. Utwórz zbiór danych (dataset) demo_eu oraz tabelę students zawierającą 20 rekordów studentów (kolumny: id – liczba całkowita, name – tekst, group_id – tekst, wartości group_id w zakresie G1–G4). Następnie, korzystając

ze Standard SQL, oblicz liczebność rekordów w każdej grupie (group_id) przy użyciu dwóch kroków: najpierw oszacuj koszty wykonania zapytania (tryb dry-run), a potem wykonaj to samo zapytanie z twardym limitem kosztu obliczeń ustawionym co najmniej na minimalny rozliczany wolumen (10 MiB). W części programistycznej użyj Pythona (pandas-gbq), aby odczytać tabelę students do DataFrame, odfiltrować jeden wybrany podzbiór (np. group_id = 'G1') i zapisać wynik jako nową tabelę students_filtered w tym samym zbiorze danych. Całość zrealizuj wyłącznie w lokalizacji EU i w Standard SQL (bez Legacy SQL). Po zakończeniu usuń tabele pomocnicze utworzone w trakcie pracy.

Oczekiwany rezultat: działający zestaw zasobów w BigQuery (dataset EU, tabela students, tabela students_filtered) oraz poprawnie wykonane zapytanie agregujące z wcześniejszym oszacowaniem kosztu (dry-run) i uruchomieniem z limitem kosztu.

Kroki przykładowego rozwiązania:

6.1. Wstęp - wybrane polecenia BigQuery (bq)

1. Lista datasetów i tabel

```
bq --location=EU ls
bq --location=EU ls bd-labs-2025-as:demo_eu
```

2. Metadane tabeli i podgląd wierszy

```
bq --location=EU show bd-labs-2025-as:demo_eu.people
bq --location=EU head -n 5 bd-labs-2025-as:demo_eu.people
```

3. Usuwanie i kopiowanie

```
bq --location=EU rm -t bd-labs-2025-as:demo_eu.people
bq --location=EU cp bd-labs-2025-as:demo_eu.people bd-labs-2025-as:demo_eu.people_copy
```

4. Eksport do pliku CSV i import z CSV

Eksport do lokalnego pliku:

```
bq --location=EU extract --destination_format=CSV bd-labs-2025-as:demo_eu.people .\people.csv
```

Import lokalnego CSV (auto-detect schematu):

bq --location=EU load --autodetect --source_format=CSV bd-labs-2025-as:demo_eu.people_from_csv .\people.csv

5. Szacowanie i limit kosztu zapytań (EU)

Szacunek bez uruchamiania (dry-run, nie nalicza opłat): zwraca tylko liczbę bajtów do przetworzenia.
bq --location=EU query --use_legacy_sql=false --dry_run "SELECT COUNT(*) FROM \bd-labs-2025-as.demo_eu.people`"`

Twardy limit danych (zatrzymuje zbyt drogie zapytania, bez naliczania kosztu):
bq --location=EU query --use_legacy_sql=false --maximum_bytes_billed=100000000 "SELECT * FROM \bd-labs-2025-as.demo_eu.people`"`

6. Poświadczenia ADC (Application Default Credentials)

Poświadczenia domyślne aplikacji ADC to mechanizm, dzięki któremu biblioteki Google automatycznie lokalizują poświadczenia w środowisku uruchomieniowym i używają ich do dostępu do usług (np. BigQuery) bez modyfikacji kodu między środowiskiem lokalnym a chmurowym (Google Cloud, b.d.-d). W pracy wyłącznie z narzędziami w terminalu (gcloud, bq) ADC nie jest wymagane; natomiast w skryptach Pythona (pandas-gbq, google-cloud-bigquery) włączenie ADC upraszcza uwierzytelnianie i eliminuje konieczność ręcznego podawania kluczy (Google Cloud, b.d.-e).
Włączenie lokalnie:
gcloud auth application-default login
polecenie zapisuje poświadczenia w profilu użytkownika, skąd automatycznie odczytują je biblioteki klienckie (Google Cloud, n.d.-e).

6.2. Instalacja Google Cloud CLI (gcloud) z narzędziem bq

1. Uruchomienie PowerShell.

2. Instalacja Cloud SDK (zawiera bq):

winget install Google.CloudSDK

3. Sprawdzenie wersji gcloud.

Zamknij i otwórz nowe okno PowerShell. Sprawdź wersję Cloud:
gcloud --version

Przykładowy wynik działania polecenia:

```
Google Cloud SDK 542.0.0
bq 2.1.24
core 2025.10.03
gcloud-crc32c 1.0.0
gsutil 5.35
```

Polecenie to wykonujemy w celu sprawdzenia, czy instalator dodał narzędzia do PATH i czy pakiet zawiera także bq i gsutil (wypisują się w tym samym bloku wersji). Jeśli pojawi się błąd (np. ExecutionPolicy), wiemy, że trzeba naprawić środowisko zanim przejdziemy dalej.
Jeżeli na Windows 10 pojawi się komunikat o zablokowanych skryptach lub problem z aktualizacją to wykonujemy:

```
winget upgrade Google.CloudSDK --include-unknown
Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy RemoteSigned
gcloud --version
```

6.3. Logowanie i konfiguracja projektu

1. Inicjalizacja

```
gcloud init
```

Logujemy się na swoje konto Google w oknie przeglądarki i akceptujemy żądane uprawnienia.

Wracamy do terminala i tworzymy nowy projekt (jeżeli automatycznie nie zacznie się tworzyć to wpisujemy: gcloud auth login oraz gcloud config set project PROJECT_ID). Nazwa projektu musi zaczynać się małą literą, dozwolone są małe litery, cyfry i myślniki, długość 6–30 znaków. Przykładowe nazwy: bd-labs-2025-as, psk-bigdata-2025.

```
gcloud projects create bd-labs-2025-as
gcloud config set project bd-labs-2025-as
```

Jeśli projekt już projekt istnieje, to można się na niego przełączyć:

```
gcloud projects list
gcloud config set project <TWÓJ_POPRAWNY_PROJECT_ID>
```

Przykładowy wynik działania polecenia:

```
    PROJECT_ID       NAME            PROJECT_NUMBER
bd-labs-2025-as  bd-labs-2025-as  594151237205
```

2. Włączenie BigQuery API

gcloud services enable bigquery.googleapis.com

6.4. Ustawienie domyślnej lokalizację EU dla bq (.bigqueryrc)

New-Item -Path $HOME\.bigqueryrc -ItemType File -Force -Value "[general]`nlocation=EU"

W celu weryfikacji tego ustawia, sprawdzamy zawartość pliku .bigqueryrc:

Get-Content $HOME\.bigqueryrc

Wynikiem polecenia powinno być:

```
[general]
location=EU
```

6.5. Test CLI (Command-Line Interface)

Wydajemy polecenia:

gcloud config list

Przykładowy wynik działania:

```
[accessibility]
screen_reader = False
[core]
account = **********@gmail.com
disable_usage_reporting = True
project = bd-labs-2025-as

Your active configuration is: [default]
```

bq query --nouse_legacy_sql "SELECT 1 AS x"

Wynikiem powinna być tabela jednokolumnowa o nagłówku x i wartości 1.

```
+---+
| x |
+---+
| 1 |
+---+
```

6.6. Tworzenie tabeli demo_eu.students (przez załadowanie CSV z autodetekcją)

Przygotuj plik CSV o nazwie: students.csv w katalogu projektu, z 20 wierszami (nagłówki: id,name,group_id):

```
id,name,group_id
1,Ada,G1
2,Bartek,G1
3,Celina,G1
4,Darek,G1
5,Ewa,G1
6,Filip,G2
7,Gosia,G2
8,Hubert,G2
9,Iga,G2
10,Jan,G2
11,Kaja,G3
12,Leon,G3
13,Maja,G3
14,Nadia,G3
15,Olek,G3
16,Paweł,G4
17,Renata,G4
18,Sara,G4
19,Tomek,G4
20,Ula,G4
```

Sprawdź, że plik powstał:

Get-Content .\students.csv | Select-Object -First 5

Objaśnienie: wykorzystano polecenie PowerShell, gdzie:
Get-Content .\students.csv - czyta zawartość pliku students.csv z bieżącego katalogu (po jednej linii).
| - przekazuje (pipeline) wynik do następnego polecenia.
Select-Object -First 5 - wybiera tylko pierwsze 5 wierszy z wejścia.

Przykładowy wynik działania:

```
id      name    group_id
1       Ada     G1
2       Bartek  G1
3       Celina  G1
4       Darek   G1
```

6.7. Tworzenie tabeli demo_eu.students

1. Załaduj CSV do tabeli z autodetekcją schematu i pominięciem nagłówka

bq --location=EU load --autodetect --source_format=CSV --skip_leading_rows=1 bd-labs-2025-as:demo_eu.students .\students.csv

Przykładowe komunikaty po wykonaniu polecenia:

```
Upload complete.
Waiting on bqjob_r51896a09859469a3_00000199d80488c7_1 ... (1s) Current status: DONE
```

2. Metadane tabeli i podgląd danch

bq --location=EU show bd-labs-2025-as:demo_eu.students

W wyniku tego polecenia otrzymamy informacje np. Schema (id: integer, - name: string, - group_id: string), Total Rows (20), Total Bytes (370), Expiration.

bq --location=EU head -n 5 bd-labs-2025-as:demo_eu.students

Przykładowy wynik działania:

```
+----+--------+----------+
| id |  name  | group_id |
```

```
+----+--------+----------+
|  1 | Ada    | G1       |
|  2 | Bartek | G1       |
|  3 | Celina | G1       |
|  4 | Darek  | G1       |
|  5 | Ewa    | G1       |
+----+--------+----------+
```

6.8. Szacunek kosztu przetworzenia zapytania

Przed przetworzeniem zapytania sprawdzamy, ile bajtów przetworzy zapytanie (pole totalBytesProcessed) bez jego uruchamiania ani naliczania opłat. W BigQuery on-demand koszt zależy od liczby przetworzonych bajtów, z zastrzeżeniem minimalnego rozliczanego wolumenu 10 MiB na zapytanie (i co najmniej 10 MiB na każdą tabelę użytą w zapytaniu). Po wykonaniu dry-run dopiero uruchamiamy to samo zapytanie z twardym limitem (--maximum_bytes_billed) ustawionym co najmniej na 10 MiB.

bq --location=EU query --use_legacy_sql=false --dry_run ‘SELECT group_id, COUNT(*) AS n FROM `bd-labs-2025-as.demo_eu.students` GROUP BY group_id’

Uwaga: w poleceniu zastosowano pojedyncze cudzysłowy ‘ i backtick `.
Przykładowy wynik działania:

```
Query successfully validated. Assuming the tables are not modified, running this query will process 80 bytes of data.
```

Interpretacja: logicznie zapytanie przetwarza około 80 bajtów danych, ale przy rozliczaniu on-demand i tak obowiązuje minimum 10 MiB na zapytanie (oraz 10 MiB na każdą tabelę użytą w zapytaniu), więc realne rozliczenie nie będzie „80 bajtów”, tylko co najmniej 10 MiB (Google Cloud, b.d.-a; Google Cloud, b.d.-b).

6.9. Przetwarzanie zapytań z limitem

1. Uruchomienie zapytania z limitem kosztu

bq --location=EU query --use_legacy_sql=false --maximum_bytes_billed=1000000 ‘SELECT group_id, COUNT(*) AS n FROM `bd-labs-2025-as.demo_eu.students` GROUP BY group_id’

Przykładowy wynik wykonania:

```
BigQuery error in query operation: Error processing job 'bd-labs-2025-
as:bqjob_r54707886e4f7559_00000199d8180c97_1': Query exceeded limit
for bytes billed: 1000000. 10485760 or higher required.
```

Komunikat mówi: ustawiono --maximum_bytes_billed=1000000 (1 MB), a BigQuery zaokrągla rozliczenie do min. 10 MiB = 10 485 760 bajtów. Dlatego prosi o co najmniej 10485760.

Wniosek: Zadziałał limit (Google Cloud, b.d.-a; Google Cloud, b.d.-b).

2. Uruchomienie zapytania z większym limitem kosztu

```
bq --location=EU query --use_legacy_sql=false --maximum_bytes_billed=10485760
'SELECT group_id, COUNT(*) AS n FROM `bd-labs-2025-as.demo_eu.students`
GROUP BY group_id'
```

Wynik działania:

```
+----------+---+
| group_id | n |
+----------+---+
| G1       | 5 |
| G2       | 5 |
| G3       | 5 |
| G4       | 5 |
+----------+---+
```

6.10. Wykonywanie zapytań BigQuery z Pythona: co dzieje się lokalnie, a co w chmurze

Dalsza część zadania dotyczy uruchamiania zapytań SQL w BigQuery z poziomu Pythona. Kod Pythona wykonywany jest lokalnie (klient), natomiast samo przetwarzanie danych odbywa się w chmurze Google w lokalizacji EU. Python (pandas-gbq lub google-cloud-bigquery) wysyła treść zapytania w Standard SQL do usługi BigQuery, uwierzytelnia się przez Application Default Credentials (ADC) i odbiera wyniki do DataFrame. Koszt i obciążenie liczone są po stronie BigQuery, lokalnie potrzebne są jedynie biblioteki i poprawna autoryzacja. Kluczowe jest ustawienie lokalizacji EU po stronie klienta oraz stosowanie dry-run do oszacowania przetwarzanych bajtów i (w razie potrzeby) limitu maximum_bytes_billed przy

uruchomieniu. Dzięki temu zachowujemy spójność lokalizacji danych, kontrolę kosztów i prostą integrację wyników z dalszym kodem analitycznym w Pythonie.

6.11. Konfiguracja środowiska lokalnego do pracy z BigQuery (Python)

1. Przejdź do katalogu projektu (PowerShell)

```
cd C:\sciezka\do\projektu
```

2. Utwórz i aktywuj środowisko wirtualne

Wirtualne środowisko Pythona venv to osobny folder z własnym interpreterem i zestawem pakietów dla danego projektu. Dzięki temu biblioteki różnych projektów się nie mieszają i łatwiej odtworzyć konfigurację.

```
py -m venv .venv
.\.venv\Scripts\Activate.ps1
```

Jako wynik polecenia można zauważyć, że teraz w lini poleceń na początku wyświetla się (.venv), np.:

```
(.venv) PS C:\AS>
```

W każdym momencie możemy sprawdzić, czy używamy właściwego interpretera poleceniem:

```
python -c "import sys; print(sys.executable)"
```

Wówczas powinno być zwrócona ścieżka z końcówką: .venv\Scripts\python.exe

3. Zainstaluj wymagane pakiety w aktywnym venv (używaj python, nie py)

```
python -m pip install --upgrade pip
python -m pip install pandas pandas-gbq pyarrow tqdm
```

4. Skonfiguruj poświadczenia ADC dla bibliotek Pythona

```
gcloud auth application-default login
```

Otworzy się okno przeglądarki domyślnej. Po udzieleniu zgody (powinno pojawić się: You are now authenticated with the gcloud CLI!) okno przeglądarki można zamknąć.

6.12. Pobieranie danych z BigQuery w Pythonie i publikacja tabeli wynikowej

1. Zapisz skrypt Pythona students_query.py w katalogu projektu

```
import pandas as pd
from pandas_gbq import read_gbq, to_gbq

PROJECT_ID = "bd-labs-2025-as"
LOCATION = "EU"

sql = """
SELECT id, name, group_id
FROM `bd-labs-2025-as.demo_eu.students`
ORDER BY id
"""

# Zapytanie wykona się w BigQuery (EU), klient działa lokalnie
df = read_gbq(sql, project_id=PROJECT_ID, location=LOCATION)

# Filtr: tylko G1
df_f = df[df["group_id"] == "G1"]

# Zapis filtrowanego wyniku do nowej tabeli w tym samym dataset
to_gbq(
    df_f,
    destination_table="demo_eu.students_filtered",
    project_id=PROJECT_ID,
    if_exists="replace",
    location=LOCATION
)

print(df_f.head())
```

2. Uruchom skrypt (wciąż w aktywnym venv)

```
python .\students_query.py
```

Przykładowy wynik działania:

```
(.venv) PS C:\Users\AS> python .\students_query.py
Downloading: 100%|                                                                |
100%|                                                                | 1/1 [00:00<?, ?it/s]
   id    name group_id
0   1     Ada       G1
1   2  Bartek       G1
2   3  Celina       G1
3   4   Darek       G1
4   5     Ewa       G1
(.venv) PS C:\Users\AS>
```

Pobieranie tabeli wynikowej na dysk lokalny

Pobrać wynik możemy na kilka sposobów. Jednym z nich jest przez kod Pythona:

```
from pandas_gbq import read_gbqdf = read_gbq(
    "SELECT * FROM `bd-labs-2025-as.demo_eu.students_filtered`",
    project_id="bd-labs-2025-as",
    location="EU"
)
df.to_csv("students_filtered.csv", index=False, encoding="utf-8")
```

Uruchom powyższy kod w środowisku venv, a wynik zostanie zapisany do pliku students_filtered.csv. Na ekranie będzie widoczny pasek postępu:

```
Downloading: 100%|                                                                |
```

6.13. Kończenie pracy

1. Posprzątaj tabele pomocnicze

```
bq --location=EU rm -t bd-labs-2025-as:demo_eu.students_filtered
bq --location=EU rm -t bd-labs-2025-as:demo_eu.students
```

2. Dezaktywuj środowisko wirtualne

```
deactivate
```

6.14. Podsumowanie - koszty

W zaprezentowanym przykładzie wykorzystano BigQuery Sandbox – tryb, w którym, przy pracy na publicznych zbiorach i w granicach bezpłatnych limitów, opłaty nie są naliczane. Po spełnieniu tych warunków koszt wynosi 0 USD (Google Cloud, 2025; Google Cloud, b.d.-a).

Poza Sandboxem opłaty dotyczą wyłącznie przetwarzania zapytań w BigQuery (model on-demand). Zapis na dysk lokalny oraz potok w PowerShell (| Out-File) są bezpłatne, koszt dotyczy jedynie pracy silnika BigQuery. Dla kontroli wydatków zaleca się użycie --maximum_bytes_billed oraz wcześniejszego dry-run (Google Cloud, b.d.-a; Google Cloud, b.d.-b; Google Cloud, b.d.-c).

Obliczanie kosztów (on-demand):

- Rozliczenie zależy od liczby bajtów przetworzonych przez zapytanie, zaokrąglanych do najbliższego MB, z minimum 10 MB na zapytanie oraz 10 MB na każdą referencjonowaną tabelę (Google Cloud, b.d.-a).
- Stawka wynosi $6.25 za 1 TB przetworzonych danych, pierwszy 1 TB/miesiąc jest bezpłatny (free tier). Po zmieszczeniu się w tym limicie koszt wynosi 0 USD (Google Cloud, b.d.-a).

Przykład (zapytanie korzysta z 1 tabeli):
Minimalnie naliczane jest 10 MB, co odpowiada ok. $0.0000625 (10 MB × $6.25 / 1 000 000 MB). Dla czterech tabel minimum wyniosłoby 40 MB ≈ $0.00025 (wyliczenie na podstawie oficjalnej stawki i reguły minimum 10 MB/tabela; Google Cloud, b.d.-a).

W przypadku uruchamiania z Pythona wymagana jest poprawna autoryzacja (ADC / gcloud auth application-default login). Sama autoryzacja nie generuje kosztów zapytań, koszty powstają dopiero podczas wykonania SQL w BigQuery (Google Cloud, b.d.-d; Google Cloud, b.d.-e; Google Cloud, b.d.-f).

7. Implementacja - Analiza publicznych danych w chmurze BigQuery

Zadanie:

Masz publiczną tabelę stacji rowerowych „bigquery-public-data.london_bicycles.cycle_stations” w BigQuery (Sandbox, lokalizacja EU). Policz, ile stacji zaczyna się na każdą literę alfabetu (agregacja po pierwszej literze nazwy

stacji). Najpierw oszacuj koszt zapytania (dry-run), potem uruchom to samo zapytanie z limitem kosztu 10 MiB. Następnie wybierz tylko te stacje, których nazwa zaczyna się na literę „A”, i zapisz wynik do tabeli „demo_eu.stations_initial_a” w swoim projekcie (EU). Na końcu podaj: liczbę liter (grup) w wyniku agregacji, liczbę wierszy zapisanych do „stations_initial_a”, wartość przetwarzanych bajtów z dry-run oraz użyty limit kosztu.

Kroki przykładowego rozwiązania:

1. Logowanie i weryfikacja konta używanego przez Cloud

gcloud auth login
gcloud auth list

Komentarz:
- gcloud auth login otwiera przeglądarkę do uwierzytelnienia,
- gcloud auth list pozwala sprawdzić, które konto ma status ACTIVE.

2. Utworzenie projektu GCP na potrzeby ćwiczenia.

cloud projects create bd-labs-2025-as-02 --name="bd-labs-2025-as-02"

Komentarz:
Jeśli projekt bd-labs-2025-as-02 już istnieje lub nie masz uprawnień do tworzenia nowych, pomiń ten krok i użyj istniejącego PROJECT_ID. Nazwa projektu jest stała po utworzeniu.

3. Ustawienie aktywnego projektu i przypisanie go jako projektu „quota/billing” dla Application Default Credentials (ADC).

gcloud config set project bd-labs-2025-as-02
gcloud auth application-default set-quota-project bd-labs-2025-as-02

Komentarz: Dzięki temu poleceniu wszystkie kolejne polecenia i biblioteki (np. bq, pandas-gbq) będą domyślnie używać właściwego projektu.

4. Kontrola konfiguracji

gcloud config list
gcloud auth list

Komentarz:
- gcloud config list powinno pokazać core.project = bd-labs-2025-as-02 oraz właściwe konto w core.account.
- gcloud auth list potwierdza, które konto jest ACTIVE.

5. Ustawienie domyślnej lokalizacji BigQuery na EU poprzez plik .bigqueryrc.

```
New-Item -Path $HOME\.bigqueryrc -ItemType File -Force -Value
"[general]`nlocation=EU"
Get-Content $HOME\.bigqueryrc
```

Komentarz: Dzięki temu nie trzeba dopisywać --location=EU w każdym poleceniu; komenda tworzy plik z sekcją [general] i kluczem location=EU. Get-Content służy do szybkiego podglądu zawartości.

6. Utworzenie datasetu demo_eu w projekcie (lokalizacja EU).

```
bq --location=EU mk --dataset bd-labs-2025-as-02:demo_eu
```

Komentarz: Jeżeli dataset już istnieje, pojawi się błąd „Already Exists”, można go zignorować lub pominąć ten krok.

7. Zliczanie stacji według pierwszej litery nazwy

Najpierw wykonujemy dry-run do oszacowania kosztu skanowania danych (bez naliczania opłat ani wykonywania obliczeń):

```
bq --location=EU query --use_legacy_sql=false --dry_run=true 'SELECT
SUBSTR(name,1,1) AS initial, COUNT(*) AS n_rows FROM `bigquery-public-
data.london_bicycles.cycle_stations` GROUP BY initial ORDER BY initial'
```

Wynik:

```
Query successfully validated. Assuming the tables are not modified, running this
query will process 23198 bytes of data.
```

Uruchomienie tego samego zapytania z „bezpiecznikiem” --maximum_bytes_billed=10485760 (10 MiB). Liczba 23198 bajtów z dry-run jest dużo poniżej limitu, więc zapytanie wykona się bez przekroczenia kosztu.

bq --location=EU query --use_legacy_sql=false --maximum_bytes_billed=10485760 'SELECT SUBSTR(name,1,1) AS initial, COUNT(*) AS n_rows FROM `bigquery-public-data.london_bicycles.cycle_stations` GROUP BY initial ORDER BY initial'

Wynikowa tabela zlicza stacje według pierwszej litery nazwy i pokazuje rozkład liczebności dla poszczególnych inicjałów.

initial	n_rows
A	40
B	76
C	75
D	20
E	30
F	31
G	48
H	47
I	9
J	4
K	18
L	37
M	31
N	30
O	12
P	48
Q	13
R	28
S	95
T	28
U	7
V	13
W	58
Y	2

8. Liczba liter (grup) w agregacji:

bq --location=EU query --use_legacy_sql=false --dry_run=true 'SELECT SUBSTR(name,1,1) AS initial, COUNT(*) AS n_rows FROM `bigquery-public-data.london_bicycles.cycle_stations` GROUP BY initial ORDER BY initial'

Query successfully validated. Assuming the tables are not modified, running this query will process 23198 bytes of data.

bq --location=EU query --use_legacy_sql=false 'SELECT COUNT(*) AS n_groups FROM (SELECT DISTINCT SUBSTR(name,1,1) AS initial FROM `bigquery-public-data.london_bicycles.cycle_stations`)'

Wynik:

```
+----------+
| n_groups |
+----------+
|       24 |
+----------+
```

Komentarz:

Najpierw powtarzamy dry-run, aby odnotować przewidywany koszt skanowania (23198 B) dla zapytania z grupowaniem po pierwszej literze. Następnie uruchamiamy krótkie zapytanie liczące liczbę unikatowych inicjałów (COUNT(*) nad SELECT DISTINCT SUBSTR(name,1,1)). Do raportu wpisujemy dwie rzeczy: liczbę grup n_groups = 24 oraz wartość przetwarzanych bajtów z dry-run (oraz użyty limit 10 MiB, który nie został przekroczony). Dodatkowa kontrola spójności: liczba wierszy z tabelarycznego wyniku agregacji (lista liter A…Y) powinna równać się n_groups.iczebności dla poszczególnych inicjałów.

9. Przygotowanie filtra dla litery ‘A’

Sprawdzamy schemat tabeli – jak dokładnie nazywają się kolumny:

bq --location=EU show --format=prettyjson bigquery-public-data:london_bicycles.cycle_stations

Wynik:


```
{
  "creationTime": "1501113916947",
  "etag": "Uwc3eEwIUV0nr+LtzdyAAg==",
  "id": "bigquery-public-data:london_bicycles.cycle_stations",
  "kind": "bigquery#table",
  "lastModifiedTime": "1749254439165",
```

```
"location": "EU",
"numActiveLogicalBytes": "0",
"numActivePhysicalBytes": "0",
"numBytes": "80084",
"numCurrentPhysicalBytes": "29325",
"numLongTermBytes": "80084",
"numLongTermLogicalBytes": "80084",
"numLongTermPhysicalBytes": "29325",
"numRows": "800",
"numTimeTravelPhysicalBytes": "0",
"numTotalLogicalBytes": "80084",
"numTotalPhysicalBytes": "29325",
"schema": {
 "fields": [
  {
   "mode": "NULLABLE",
   "name": "id",
   "type": "INTEGER"
  },
  {
   "mode": "NULLABLE",
   "name": "installed",
   "type": "BOOLEAN"
  },
  {
   "mode": "NULLABLE",
   "name": "latitude",
   "type": "FLOAT"
  },
  {
   "mode": "NULLABLE",
   "name": "locked",
   "type": "STRING"
  },
  {
   "mode": "NULLABLE",
   "name": "longitude",
   "type": "FLOAT"
  },
  {
   "mode": "NULLABLE",
```

```
    "name": "name",
    "type": "STRING"
   },
   {
    "mode": "NULLABLE",
    "name": "bikes_count",
    "type": "INTEGER"
   },
   {
    "mode": "NULLABLE",
    "name": "docks_count",
    "type": "INTEGER"
   },
   {
    "mode": "NULLABLE",
    "name": "nbEmptyDocks",
    "type": "INTEGER"
   },
   {
    "mode": "NULLABLE",
    "name": "temporary",
    "type": "BOOLEAN"
   },
   {
    "mode": "NULLABLE",
    "name": "terminal_name",
    "type": "STRING"
   },
   {
    "mode": "NULLABLE",
    "name": "install_date",
    "type": "DATE"
   },
   {
    "mode": "NULLABLE",
    "name": "removal_date",
    "type": "DATE"
   }
  ]
 },
```

```
    "selfLink": "https://bigquery.googleapis.com/bigquery/v2/projects/bigquery-
      public-data/datasets/london_bicycles/tables/cycle_stations",
    "tableReference": {
     "datasetId": "london_bicycles",
     "projectId": "bigquery-public-data",
     "tableId": "cycle_stations"
    },
    "type": "TABLE"
}
```


Inny sposób sprawdzenia schematu tabeli:

bq --location=EU show bigquery-public-data:london_bicycles.cycle_stations

Część tabeli zwróconej w wyniku:

|- id: integer
|- installed: boolean
|- latitude: float
|- locked: string
|- longitude: float
|- name: string
|- bikes_count: integer
|- docks_count: integer
|- nbEmptyDocks: integer
|- temporary: boolean
|- terminal_name: string
|- install_date: date
|- removal_date: date

Komentarz:

Zanim zapiszemy wynik do swojej tabeli, sprawdzamy schemat źródłowej tabeli, aby mieć pewność co do nazw i typów kolumn (tu kluczowe: name jako STRING). Pokazujemy dwie formy podglądu schematu (JSON i skrót wierszowy), co ułatwia później budowanie zapytań i weryfikację wyników.

10. Najprostsza wersja zapytania „A”

Zapisz do pliku q_a.sql zapytanie SQL: (w Notatniku zapisz plik w UTF-8)

```
SELECT *
FROM `bigquery-public-data.london_bicycles.cycle_stations`
WHERE SUBSTR(name,1,1) = 'A';
```

Dry-run:

```
Get-Content -Raw .\q_a.sql | bq --location=EU query --use_legacy_sql=false --
dry_run=true
```

Wynik:

```
Query successfully validated. Assuming the tables are not modified, running this
query will process 80084 bytes of data.
```

Wykonanie z limitem 10 MiB i zapis do datasetu (EU)

```
Get-Content -Raw .\q_a.sql | bq --location=EU query --use_legacy_sql=false --
maximum_bytes_billed=10485760 --destination_table=bd-labs-2025-as-
02:demo_eu.stations_initial_a --replace
```

Omówienie komendy:

- BigQuery uruchamia zapytanie i zapisuje wynik do tabeli bd-labs-2025-as-02:demo_eu.stations_initial_a w EU.
- Tabela zostaje nadpisana (bo --replace) i zawiera tylko stacje, których name zaczyna się na „A”.

Wynik: na ekranie wyświetli się tabela ze wszystkimi kolumnami dla wszystkich wierszy ze stacjami zaczynającymi się od A

Komentarz:

Definiujemy najprostszy filtr: w pliku q_a.sql wybieramy wszystkie kolumny z wierszy, gdzie SUBSTR(name,1,1) = 'A' (alternatywnie można użyć STARTS_WITH(name,'A')). Po pozytywnym dry-runie uruchamiamy zapis wyniku do demo_eu.stations_initial_a z --maximum_bytes_billed=10485760 i --replace (lokalizacja EU). W raporcie odnotowujemy liczbę zapisanych wierszy (40) oraz że wynikowa tabela zawiera pełny schemat źródła, a jedynie ograniczony zestaw wierszy.

11. Weryfikacja liczby zapisanych wierszy:

Do pliku q_count.sql zapisujemy:

```
SELECT COUNT(*) AS rows_in_a
FROM `bd-labs-2025-as-02.demo_eu.stations_initial_a`;
```

W PowerShell uruchamiamy:

```
Get-Content -Raw .\q_count.sql | bq --location=EU query --use_legacy_sql=false
```

Wynik:

```
+-----------+
| rows_in_a |
+-----------+
|        40 |
+-----------+
```

8. Implementacja - Podsumowanie

8.1. Cel rozdziałów 6-7

Rozdziały 6 i 7 pokazują praktyczny przegląd pracy z BigQuery w regionie EU: od zbudowania własnego, niewielkiego zbioru danych i wykonania pierwszych zapytań po analizę gotowych, publicznych tabel dostępnych bezpośrednio w BigQuery. Celem było opanowanie podstaw narzędzi (gcloud, bq, Python/pandas-gbq), zrozumienie modelu kosztowego oraz wyrobienie nawyku pracy w Standard SQL z pełną kontrolą lokalizacji i parametrów wykonania.

8.2. Różnica pomiędzy projektami

W rozdziale 8 zbudowano własny zbiór danych, przećwiczono podstawy pracy z CLI i Pythonem oraz konsekwentnie stosowano mechanizmy kontroli kosztów (dry-run i maximum_bytes_billed). W rozdziale 9 pracowano już na publicznych danych w BigQuery, koncentrując się na zapytaniach agregujących i materializacji wyników do tabel w projekcie, aby uzyskać trwały, powtarzalny rezultat analizy.

8.3. Wspólne wzorce i dobre praktyki

W obu rozdziałach kluczowe jest ustawienie lokalizacji EU oraz używanie Standard SQL. Każde zapytanie powinno być poprzedzone dry-runem, a właściwe wykonanie uruchamiane z limitem maximum_bytes_billed (co najmniej 10 MiB), aby zapobiec niekontrolowanym kosztom. Wyniki materializuje się wyłącznie wtedy, gdy przynosi to korzyści (wersjonowanie, szybkie ponowne użycie, stabilizacja raportów). Po zakończeniu pracy usuwa się zasoby tymczasowe, a całość dokumentuje się starannie: projekt, dataset, lokalizację, treść zapytań, liczniki wyników (np. liczba grup, liczba wierszy po filtrze) oraz wartości z dry-run i zastosowane limity.

9. Krótkie wyjaśnienia zastosowanych pojęć

1 MiB - mebibajt, jednostka binarna. 1 MiB = 1 048 576 bajtów (1024 × 1024)

CLI, Command-Line Interface - interfejs wiersza poleceń

DCAT-AP (Data Catalog Vocabulary - Application Profile) to europejski profil słownika W3C DCAT*, który ujednolica metadane zbiorów danych publicznych, by ułatwić wymianę opisów i wyszukiwanie między portalami. Definiuje podstawowe elementy (katalog, zbiór, dystrybucja, usługa danych) oraz minimalne wymagania ich opisu, zapewniając interoperacyjność w administracji.

DOI, Digital Object Identifier - rejestrowany m.in. przez DataCite, trwały identyfikator cyfrowy zapewniający stały link do zasobu i jednoznaczne cytowanie zbioru danych

Parquet - otwarty, kolumnowy format plików do wydajnego przechowywania i odczytu danych (projekt Apache). Zapewnia silną kompresję i kodowanie kolumn, zapis schematu w pliku oraz statystyki umożliwiające pomijanie niepotrzebnych fragmentów podczas zapytań (predicate pushdown). Jest szeroko wspierany w narzędziach analitycznych (np. Spark, narzędzia chmurowe) (Apache Parquet, 2025).

Pip - W Pythonie pip to narzędzie służące do instalowania i zarządzania pakietami (czyli bibliotekami) z repozytorium Python Package Index (PyPI).

W3C DCAT (Data Catalog Vocabulary) to słownik służący do opisu katalogów danych w sieci, tak aby metadane były interoperacyjne i łatwe do agregowania oraz wyszukiwania między portalami.

10. Literatura

- ✓ materiały zwarte:
  - Jaruga, D. (2021). Komunikacja sieciowa. Źródła informacji Big Data. Wydawnictwo Naukowe i Edukacyjne Stowarzyszenia Bibliotekarzy Polskich.
  - Pawłowska, M. M., & Wachowicz, M. E. (2020) Wprowadzenie do zarządzania danymi naukowymi, Wydawnictwo Difin.
  - Szudejko, M. (2024). Storytelling oparty na danych. Helion. Wydawnictwo Helion.
- ✓ artykuły naukowe:
  - Wilk-Jakubowski, J. L., Pawlik, L., Wilk-Jakubowski, G., & Sikora, A. (2025). Machine Learning and Neural Networks for Phishing Detection: A Systematic Review (2017–2024). Electronics, 14(18), 3744. https://doi.org/10.3390/electronics14183744
  - Wilk-Jakubowski, J. Ł., Sikora, A., & Maciejski, D. (2025). The effectiveness of machine learning in detecting phishing websites. Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Środowiska, 15(3), 105–109. https://doi.org/10.35784/iapgos.8143
  - Vanschoren, J., van Rijn, J. N., Bischl, B., & Torgo, L. (2014). OpenML: Networked science in machine learning. SIGKDD Explorations, 15(2), 49–60. https://arxiv.org/abs/1407.7722
- ✓ źródła internetowe:
  - Amazon Web Services. (b.d.-a). Registry of Open Data on AWS. (https://registry.opendata.aws/, dostępny 11.09.2025)
  - Amazon Web Services. (b.d.-b). Amazon Athena — pricing. (https://aws.amazon.com/athena/pricing/, dostępny 29.09.2025)
  - Apache Parquet. (2025). Overview. Apache Software Foundation. https://parquet.apache.org/docs/overview/
  - DataCite Metadata Working Group. (2024). DataCite Metadata Schema Documentation for the Publication and Citation of Research Data, Version 4.5. https://doi.org/10.14454/g8e5-6293
  - European Environment Agency. (b.d.). European Environment Agency. (https://www.eea.europa.eu/en, dostępny 11.09.2025)
  - Eurostat. (b.d.-a). API - Getting started. (https://ec.europa.eu/eurostat/web/user-guides/data-browser/api-data-access/api-getting-started, dostępny 11.09.2025)
  - Eurostat. (b.d.-b). Data Browser — user guides. (https://ec.europa.eu/eurostat/web/user-guides/data-browser, dostępny 11.09.2025)

- Eurostat. (b.d.-c). Home - Eurostat. (https://ec.europa.eu/eurostat, dostępny 11.09.2025)
- Generalna Dyrekcja Dróg Krajowych i Autostrad. (d.b.). Generalny Pomiar Ruchu 2025. (https://www.gov.pl/web/gddkia/generalny-pomiar-ruchu-2025, dostępny 11.09.2025)
- Generalna Dyrekcja Dróg Krajowych i Autostrad. (d.b.). Stacje Ciągłych Pomiarów Ruchu (SCPR). (https://www.gov.pl/web/gddkia/stacje-ciaglych-pomiarow-ruchu, dostępny 11.09.2025)
- Główny Urząd Geodezji i Kartografii. (d.b.). Geoportal.gov.pl – Geoportal Infrastruktury Informacji Przestrzennej. (https://www.geoportal.gov.pl/, dostępny 11.09.2025)
- Główny Urząd Statystyczny. (b.d.). Bank Danych Lokalnych (BDL). (https://bdl.stat.gov.pl/, dostępny 11.09.2025)
- Główny Urząd Statystyczny. (2018, 4 grudnia). API BDL - dokumentacja REST. https://api.stat.gov.pl/Home/BdlApi
- Google Cloud. (2025, October 2). BigQuery public datasets. https://cloud.google.com/bigquery/public-data
- Google Cloud. (b.d.-a). BigQuery – pricing. (https://cloud.google.com/bigquery/pricing, dostępny 29.09.2025)
- Google Cloud. (b.d.-b). BigQuery – estimate and control costs (best practices). (https://cloud.google.com/bigquery/docs/best-practices-costs, dostępny 29.09.2025)
- Google Cloud. (b.d.-c). Dry run query. (https://cloud.google.com/bigquery/docs/samples/bigquery-query-dry-run, dostępny 29.09.2025)
- Google Cloud. (b.d.-d). How Application Default Credentials works. (https://cloud.google.com/docs/authentication/application-default-credentials, dostępny 29.09.2025).
- Google Cloud. (b.d.-e). gcloud auth application-default login (CLI reference) (https://cloud.google.com/sdk/gcloud/reference/auth/application-default/login, dostępny 29.09.2025).
- Google Cloud. (b.d.-f). Authenticate to BigQuery. (https://cloud.google.com/bigquery/docs/authentication, dostępny 11.09.2025).
- Kaggle. (b.d.-a). Kaggle. (https://www.kaggle.com/, dostępny 01.09.2025)
- Kaggle. (b.d.-b). Datasets. (https://www.kaggle.com/datasets, dostępny 01.09.2025)

- Kaggle. (b.d.-c). Notebooks. (https://www.kaggle.com/code, dostępny 01.09.2025)
- Kaggle. (b.d.-d). Kaggle API. (https://github.com/Kaggle/kaggle-api, dostępny 01.09.2025)
- Kelly, M., Longjohn, R., & Nottingham, K. (b.d.). The UCI Machine Learning Repository. (https://archive.ics.uci.edu, dostępny 22.09.2025
- Mendeley. (b.d.). Mendeley Data. (https://data.mendeley.com/, dostępny 03.09.2025)
- Microsoft. (2025, 12 września). What are Azure Open Datasets and how can you use them? https://learn.microsoft.com/en-us/azure/open-datasets/overview-what-are-open-datasets
- Microsoft. (b.d.). Azure Open Datasets. (https://azure.microsoft.com/en-us/products/open-datasets, dostępny 03.09.2025)
- Ministerstwo Cyfryzacji. (b.d.). Otwarte Dane - o portalu. (https://dane.gov.pl/pl/page/o-serwisie?lang=pl, dostępny 29.09.2025)
- OpenML. (b.d.). OpenML. (https://www.openml.org/, dostępny 29.09.2025)
- Publications Office of the European Union. (b.d.-a). About data.europa.eu. (https://data.europa.eu/en/about/about-dataeuropaeu, dostępny 10.09.2025)
- Publications Office of the European Union. (b.d.-b). data.europa.eu Academy. (https://data.europa.eu/en/academy, dostępny 29.09.2025)
- Publications Office of the European Union. (b.d.-c). DCAT-AP for data portals in Europe. (https://op.europa.eu/en/web/eu-vocabularies/dcat-ap, dostępny 11.09.2025)
- Venema, W., & Polley, G. (2023, 23 lutego). Building streaming data pipelines on Google Cloud. Google Cloud Blog. https://cloud.google.com/blog/products/data-analytics/building-streaming-data-pipelines