Łukasz Prokulski

Jak zostać analitykiem danych?

Jak używać tej książki?

Niniejszy przewodnik przedstawia kompleksową ścieżkę rozwoju od absolutnych podstaw do profesjonalnej kariery w analizie danych.

Traktuj go jak mapę – listę rzeczy do nauczenia się, poszukania i zgłębienia na własną rękę. To nie jest kurs, to kompas w świecie analizy danych.

Tej książki nie musisz czytać od początku do końca. Możesz sięgać po wybrane rozdziały – i właśnie tak zakładam, że będziesz postępować. Część zagadnień już znasz, część może Cię nie interesować lub nigdy nie będzie potrzebna w Twojej ścieżce zawodowej. Jeśli chcesz tworzyć wykresy w Power BI, niekoniecznie musisz umieć stawiać klaster przetwarzający dane w Google Cloud Platform.

To jest Twoja mapa

- Jeśli zabłądzisz – użyj mapy
- Jeśli zastanawiasz się nad kolejnym kierunkiem rozwoju – zerknij do mapy i wybierz nową trasę

To jest **ściągawka na Twojej drodze rozwoju** – Twój osobisty plan kariery.

Z dobrym planem, konsekwentnym wysiłkiem i zamiłowaniem do uczenia się możesz zbudować satysfakcjonującą karierę w świecie analizy danych, nawet zaczynając od zera.

Spis treści

Początek

Excel

SQL

Statystyka

Python

Architektura rozwiązań

Data Scientist

Data Engineer

BI Analyst / Analityk biznesowy

Umiejętności miękkie

Zakończenie

Bonus

Wstęp

Najlepiej zacząć od

Cześć!

Przed Tobą książka o tym, jak zostać analitykiem danych albo - co bardziej prawdziwe - jak zdobyć wiedzę, aby pracować z danymi. Bo z danymi można pracować na różnych stanowiskach, nie tylko jako *analityk danych*.

Możesz być specjalistą (lub specjalistką) od wykresów i dashboardów, czyli **analitykiem BI** (z angielskiego *BI analyst*).

Możesz tworzyć modele, które wyliczają czy na obrazku mamy psa czy kota, albo kiedy zabraknie prądu, albo czy klient banku spłaci kredyt - wówczas jesteś bardziej **data scientistą** (z braku lepszego polskiego określenia; niektórzy mówią *danolog*).

Ale oba wspomniane stanowiska to mogą być Twoje koleżanki i Twoi koledzy w pracy, a Ty będziesz im danych dostarczać i sprawiać, żeby było im wygodnie, a wszystko było uporządkowało i działało poprawnie. Wówczas będziesz **inżynierem danych**, czyli po angielsku *data engineer*.

Autor

Zanim zaczniemy, pozwól, że się krótko przedstawię:

- Od co najmniej 25 lat *pracuję w IT* – część tego czasu to jeszcze studia i średnia szkoła, nie tylko *prawdziwa* praca.
- Pisałem w kilku językach programowania (między innymi w C, PHP, R, Python i SQL)
- **Około 15 lat zajmuję się analizą danych** – stworzyłem setki, jeśli nie tysiące analiz, raportów, sporo dashboardów, kilka modeli predykcyjnych, kilka przepływów danych. To wszystko pisząc formuły w Excelu, kod w VBA, R i Pythonie, *brudząc sobie ręce*. Drugie tyle rozwiązań zaprojektowałem - od kilku lat moja główna praca to projektowania rozwiązań związanych z danymi, a nawet big data.
- Nie lubię pracy powtarzalnej, dlatego zaprojektowałem (i część z nich własnoręcznie wdrożyłem) rozwiązania do przetwarzania danych.
- W tym automaty generujące raporty oraz aplikacje zbierające i prezentujące dane.
- To były projekty przygotowane dla różnych zespołów na różnych szczeblach organizacji, w tym dla prezesów spółek giełdowych z WIG30. Zarządom i dyrektorom tych spółek prezentowałem dane - najczęściej jako raporty czy gotowe dashboardy. Ustalałem z nimi jak mają wyglądać.

Do tego wszystkiego lubię dzielić się wiedzą:

- Chcę przekazać jej trochę **Tobie**.
- Stworzyłem pierwszego w Polsce bloga o analizie danych, gdzie o analizie danych pisałem od lutego 2017 roku (intensywnie do połowy 2022)
- Prowadzę fanpage poświęcony tym zagadnieniom. Jeśli trzymać się dat - od lipca 2017 roku.

- Oraz newsletter (też chyba pierwszy w Polsce, pierwszy numer wyszedł na Boże Narodzenie 2018 roku) z ręcznie wybranymi materiałami na ten temat.
- Prowadzę szkolenia – głównie o Pythonie i analizie danych.
- Na początku 2025 roku napisałem e-booka z gotowym projektem w Pythonie, którego celem jest przeprowadzenie czytelnika z etapu **junior** do **mid** (**kup go!**)
- Jakoś w marcu 2025 roku stworzyłem darmowy cykl prezentujący typowy projekt analityczny, dający na koniec model predykcyjny - **zapisz się na pakiet lekcji o analizie rynku mieszkaniowego**

Mam nadzieję, że to wystarczające powody, aby mi zaufać. Wiedzę przekazuję jak widzisz od ponad 8 lat.

Zaczynajmy

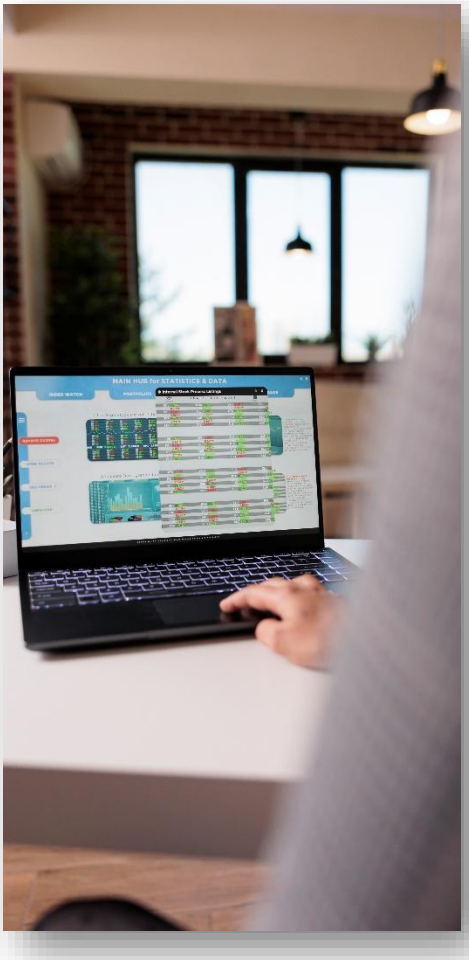
W maju 2017 r. *The Economist* opublikował artykuł zatytułowany "Najcenniejszym zasobem świata nie jest już ropa, ale dane". Od czasu publikacji temat wywołał wiele dyskusji, a **Dane to nowa ropa** (ang. *data is new oil*) stało się powszechnym refrenem. Od analizy danych przez data science po inżynierię danych - zawody związane z przetwarzaniem i analizą danych liczbowych oferują fascynujące możliwości dla osób, które lubią pracę z liczbami. "Branża danych" przeżywa ciągle dynamiczny rozwój, z prognozowanym wzrostem liczby miejsc pracy o 35% do 2032 roku według amerykańskiego Biura Statystyki Pracy. Nawet bez doświadczenia w Excelu czy programowaniu, możesz zbudować karierę w tej dziedzinie poprzez systematyczne nabywanie umiejętności i praktyki.

Excel - król korporacji

...a za punkt zero uznać można Excela. Program-kombajn, program-legenda. Najbardziej znany arkusz kalkulacyjny świata (chociaż są inne - chociażby Google Sheets). Program, bez którego żadna korporacja nie potrafi istnieć. Pokusiłbym się nawet o parafrazę: "Excel jest król korporacji, jak szczupak jest król wód, a lew król dżungli!" **Trzeba go więc znać**. Oraz Power Pointa.

Excela uczą już w szkołach podstawowych, ale wówczas ważniejsze jest zdobywanie pucharków w Brawl Stars, budowanie domków w The Sims albo (miejmy nadzieję, że wciąż) rower, kto wyjdzie wyżej na drzewo, albo czyją *bazę* dzisiaj zdobywamy (taką ze starych opon i patyków, oczywiście).

Zakładam więc, że nie potrafisz niczego w Excelu. Proponowany przeze mnie plan nauki tego narzędzia obejmuje kilka etapów:



- **Etap 1: Podstawy** - nawigacja po samym programie, typy danych, adresowanie komórek.
- **Etap 2: Formatowanie i organizacja danych** - formatowanie komórek, sortowanie, filtrowanie, usuwanie duplikatów, zamrażanie okienek.
- **Etap 3: Podstawowe formuły** - funkcje matematyczne, statystyczne, logiczne, tekstowe, podstawy funkcji wyszukiwania jak WYSZUKAJ.PIONOWO.
- **Etap 4: Wizualizacja danych** - tworzenie podstawowych wykresów: kolumnowych, liniowych, kołowych, punktowych.
- **Etap 5: Zaawansowane narzędzia** - tabele przestawne, zaawansowane formatowanie warunkowe.
- **Etap 6: Import i przetwarzanie danych** - wczytywanie danych z plików CSV, walidacja i czyszczenie danych.
- **Etap 7: Zaawansowane funkcje** - zaawansowane funkcje wyszukiwania, funkcje agregujące z warunkami, funkcje pracy z datami.

Tak wygląda to w dużym skrócie, w widoku z góry. Jeśli wszystkie te elementy są Ci doskonale znane to możesz spokojnie pominąć następny rozdział - proponuję przejść do prostego quizu, aby sprawdzić czy na pewno wszystko jest jasne.

Excel - plan nauki

Jeśli jednak masz poczucie, że czegoś nie wiesz - przejrzyj detaliczny plan poniżej.

Etap 1: Podstawy (niezbędne minimum)

Dlaczego? Żeby w ogóle zrozumieć jak działa ten program. Bez niego niczego nie zrobisz w pracy, nawet jeśli nie chcesz z Excelem pracować.

1. Nawigacja i podstawowe operacje

- Poruszanie się po arkuszu (klawisze strzałek, Ctrl+strzałki, Page Up/Down)
- Zaznaczanie komórek i zakresów (myszką, klawiszami strzałek i Shiftem, mysz i Ctrl)
- Kopiowanie, wklejanie, wycinanie (Ctrl+C, Ctrl+V, Ctrl+X)
- Cofanie operacji (Ctrl+Z)

2. Typy danych i ich rozpoznawanie

- Tekst vs liczby - jak Excel interpretuje dane
- Daty i ich formaty
- Błędy (####, #DIV/0!, #NAZWA?, #N/D)
- Automatyczne wykrywanie i konwersja typów

3. Adresowanie komórek

- Względne (A1, B2)
- Bezwzględne (A1, B2)
- Mieszane ($A1, A$1)
- Nazwy zakresów

Etap 2: Formatowanie i organizacja danych

Dlaczego? Żeby Twoje tabelki wyglądały ładnie i czytelnie. Jak jest ładnie to wygląda profesjonalnie (o ile nie przesadzisz z pstrokatymi kolorami - raczej pastele, raczej tylko kilka kolorów lub ich odcieni) i że ktoś się napracował. Jeśli jest czytelne, to jest (łatwiej) zrozumiałe.

4. Formatowanie komórek

- Kolory tekstu i tła
- Ramki i obramowania

- Formatowanie liczb (waluta, procenty, miejsca po przecinku)
- Formatowanie dat
- Formatowanie warunkowe (podstawy)

5. Organizacja danych

- Zamrażanie okienek
- Sortowanie danych
- Filtrowanie danych (autofiltr)
- Usuwanie duplikatów

Etap 3: Podstawowe formuły (najważniejsze funkcje)

Dlaczego? Bo nie chcesz wszystkiego robić ręcznie. Excel to formuły, bez nich nie ma sensu. Oraz formuły (i sposób działania) nauczy Cię myślenia algorytmicznego, a bez niego zapomnij o pracy z danymi.

6. Funkcje matematyczne i statystyczne

- **SUMA** - sumowanie zakresów
- **ŚREDNIA** - średnia arytmetyczna
- **MIN/MAX** - wartości minimalne i maksymalne
- **LICZBA** - zliczanie komórek z liczbami
- **LICZ.JEŻELI** - zliczanie z warunkiem

7. Funkcje logiczne

- **JEŻELI** - podstawowa logika warunkowa
- **I/LUB** - operatory logiczne
- **CZY.PUSTE** - sprawdzanie pustych komórek

8. Funkcje tekstowe

- **POŁĄCZ** lub **&** - łączenie tekstów
- **TEKST** - formatowanie wartości jako tekst
- **LEWY/PRAWY/FRAGMENT.TEKSTU** - wyciąganie części tekstu
- **DŁ** - liczba znaków
- **LITERY.WIELKIE/LITERY.MAŁE/Z.WIELKIEJ.LITERY** - zmiana wielkości liter

9. Funkcje wyszukiwania (podstawy)

- **WYSZUKAJ.PIONOWO (VLOOKUP)** - podstawowe wyszukiwanie

- **INDEKS + PODAJ.POZYCJĘ** - bardziej elastyczne wyszukiwanie

Etap 4: Wizualizacja danych

Dlaczego? Bo jeden obraz to milion słów. Po co patrzeć na 10 wierszy w tabelce liczb jeśli wystarczy wykres i wiadomo czy nasz biznes rośnie?

10. Wykresy - najczęściej używane typy

- **Wykres kolumnowy** - porównywanie wartości
- **Wykres liniowy** - trendy w czasie
- **Wykres kołowy** - proporcje i udziały
- **Wykres punktowy (scatter)** - korelacje
- Elementy wykresu: tytuł, osie, legenda, etykiety danych
- Serie danych - co to i jak nimi zarządzać

Etap 5: Zaawansowane narzędzia

Dlaczego? Bo tutaj zaczyna się Twoja przewaga. To już robi się *Excel dla zaawansowanych*.

11. Tabele przestawne

- Tworzenie tabeli przestawnej
- Pola: wiersze, kolumny, wartości, filtry
- Grupowanie danych (daty, liczby)
- Formatowanie i odświeżanie danych
- Wykresy przestawne

12. Formatowanie warunkowe (zaawansowane)

- Skale kolorów
- Paski danych
- Zestawy ikon
- Własne reguły

Etap 6: Import i przetwarzanie danych

Dlaczego? Bo dane są w różnych formatach i ten, kto umie je sprawnie zaimportować i później sprawnie przetworzyć wygrywa.

13. Wczytywanie danych

- Pliki CSV (problemy z separatorami i kodowaniem)
- Kopiowanie z innych aplikacji
- Podstawy Power Query
- Łączenie arkuszy i skoroszytów

14. Walidacja i czyszczenie danych

- Walidacja danych (listy rozwijane, zakresy wartości)
- Zamiana tekst na kolumny
- Usuwanie spacji i znaków specjalnych
- Znajdź i zamień
- Usuwanie duplikatów

Etap 7: Zaawansowane funkcje

Dlaczego? Bo z tymi (i nie tylko tymi - szukaj więcej) funkcjami można robić czary. Kiedyś zrobiłem właściwie bilansowanie zasobów znane z MS Project w Excelu… tłumaczyłem to konsultantom z firmy z *Wielkiej Czwórki* godzinami.

15. Funkcje agregujące z warunkami

- **SUMA.JEŻELI/SUMA.WARUNKI**
- **ŚREDNIA.JEŻELI**
- **MAKS.WARUNKÓW/MIN.WARUNKÓW**

16. Funkcje pracy z datami

- **DZIŚ/TERAZ** - bieżąca data/czas
- **DZIEŃ/MIESIĄC/ROK** - wyciąganie części daty
- Obliczanie różnic w datach

Praktyczne wskazówki do nauki

- **Zacznij od małych projektów** - proste budżety domowe, listy zakupów
- **Ćwicz na prawdziwych danych** - znajdź dane, które Cię interesują
- **Używaj F1** - pomoc Excela jest bardzo dobra
- **Naciśnij F2** - aby edytować formułę w komórce
- **Ctrl+Shift+Enter** - dla formuł tablicowych (w starszych wersjach)
- **Alt+Enter** - nowa linia w tej samej komórce

Kolejne kroki (po opanowaniu podstaw)

- Power Pivot
- Makra VBA
- Power BI
- Zaawansowane funkcje Power Query

Excel - ćwiczenia

Etap 1: Podstawy

Ćwiczenie 1: Stwórz listę zakupów

Wprowadź nazwy produktów, ilości i ceny w osobnych kolumnach. Poćwicz poruszanie się po arkuszu (klawisze strzałek, Ctrl+strzałki, Page Up/Down), zaznaczanie komórek i zakresów (myszką, Shiftem, Ctrl) oraz kopiowanie, wklejanie i wycinanie danych (Ctrl+C, Ctrl+V, Ctrl+X).

Ćwiczenie 2: Utwórz prosty budżet domowy

Wpisz kategorie wydatków (np. czynsz, jedzenie, transport) i przypisane im kwoty. Zwróć uwagę na to, jak Excel interpretuje typy danych (tekst vs liczby, daty) i jak radzi sobie z błędami (#DIV/0!, #NAZWA?).

Ćwiczenie 3: Zorganizuj listę kontaktów

Stwórz tabelę z imionami, nazwiskami i numerami telefonów. W jednej z komórek wpisz fikcyjny "numer kierunkowy kraju" i zastosuj **adresowanie bezwzględne** (A1) do tej komórki, a następnie skopiuj ją, aby zobaczyć, jak adresowanie zmienia się (lub nie) przy kopiowaniu formuł.

Etap 2: Formatowanie i organizacja danych

Ćwiczenie 1: Popraw czytelność budżetu domowego

Zastosuj **kolory tekstu i tła**, ramki i obramowania, a także sformatuj liczby jako walutę z odpowiednią liczbą miejsc po przecinku. Spróbuj **zamrozić okienka**, aby nagłówki kolumn były zawsze widoczne podczas przewijania.

Ćwiczenie 2: Posortuj i filtruj listę kontaktów

Posortuj kontakty **alfabetycznie według nazwiska**, a następnie według imienia. Zastosuj **filtrowanie danych** (autofiltr), aby wyświetlić tylko kontakty z określoną literą w imieniu lub z konkretnego miasta.

Ćwiczenie 3: Oczyść listę produktów

Wymyśl listę produktów, która celowo zawiera duplikaty. Poćwicz **usuwanie duplikatów**.

Ćwiczenie 4: Wizualizuj dane dotyczące frekwencji

Stwórz fikcyjną listę frekwencji uczniów i użyj podstawowego **formatowania warunkowego** (np. czerwone tło dla nieobecnych, zielone dla obecnych).

Etap 3: Podstawowe formuły

Ćwiczenie 1: Oblicz sumę i średnią miesięcznych wydatków

Wykorzystaj funkcje **SUMA** i **ŚREDNIA** w swoim budżecie domowym. Wylicz również **MIN** i **MAX** wydatków.

Ćwiczenie 2: Sprawdź status produktu

Użyj funkcji **JEŻELI** i **CZY.PUSTE**, aby na podstawie stanu magazynowego wyświetlić w nowej kolumnie "Dostępny" lub "Brak w magazynie".

Ćwiczenie 3: Połącz dane personalne

Utwórz nową kolumnę, która łączy imię i nazwisko z listy kontaktów za pomocą funkcji **POŁĄCZ** lub operatora **&**. Poćwicz także użycie funkcji **DŁ** i **FRAGMENT.TEKSTU** do wyodrębniania części tekstu.

Ćwiczenie 4: Wyszukaj cenę produktu

Stwórz małą tabelę z produktami i ich cenami. Następnie, korzystając z funkcji **WYSZUKAJ.PIONOWO (VLOOKUP)**, wyszukaj cenę wybranego produktu. Spróbuj też użyć **INDEKS + PODAJ.POZYCJĘ** dla bardziej elastycznego wyszukiwania.

Etap 4: Wizualizacja danych

Ćwiczenie 1: Stwórz wykres słupkowy miesięcznych wydatków

Pokaż wizualnie, na co wydajesz najwięcej pieniędzy w budżecie domowym.

Ćwiczenie 2: Narysuj wykres liniowy zmian temperatury w ciągu tygodnia

Zidentyfikuj **tytuł, osie (X i Y) i legendę** wykresu. Zmień kolory i style linii.

Ćwiczenie 3: Przedstaw udziały w torcie

Jeśli masz dane o sprzedaży różnych kategorii produktów, stwórz **wykres kołowy**, aby pokazać ich proporcje.

Ćwiczenie 4: Analizuj korelację

Jeśli masz dane o godzinach nauki i wynikach egzaminów, stwórz **wykres punktowy (scatter plot)**, aby zobaczyć, czy istnieje związek między tymi dwoma zmiennymi.

Etap 5: Zaawansowane narzędzia

Ćwiczenie 1: Zbuduj tabelę przestawną z danymi sprzedażowymi

Jeśli masz listę transakcji (produkt, ilość, cena, data, region), **utwórz tabelę przestawną**, która pokaże sumę sprzedaży dla każdego produktu lub miesiąca. Poćwicz **grupowanie danych** (np. daty według miesięcy) i dodawanie różnych pól (wiersze, kolumny, wartości, filtry).

Ćwiczenie 2: Wizualizuj ranking sprzedawców za pomocą zaawansowanego formatowania warunkowego

Użyj **pasków danych**, **skal kolorów** lub **zestawów ikon**, aby szybko zobaczyć, którzy sprzedawcy osiągają najlepsze wyniki na liście wyników.

Ćwiczenie 3: Stwórz wykres przestawny

Na podstawie tabeli przestawnej z ćwiczenia 1, wygeneruj **wykres przestawny**, aby dynamicznie zmieniać widok danych przy użyciu filtrów tabeli przestawnej.

Etap 6: Import i przetwarzanie danych

Ćwiczenie 1: Wczytaj dane z pliku CSV z problemami

Znajdź plik CSV z danymi (np. otwarte dane publiczne, jak ceny akcji), który ma problem z separatorami (np. średnik zamiast przecinka) lub kodowaniem znaków. **Wczytaj go do Excela**, ręcznie poprawiając te problemy.

Ćwiczenie 2: Uporządkuj dane za pomocą “Tekst jako kolumny”

Jeśli w pobranych danych są kolumny z połączonymi wartościami (np. “Nazwa produktu - Kod produktu”), użyj narzędzia **“Tekst jako kolumny”** do ich rozdzielenia na osobne kolumny.

Ćwiczenie 3: Zwaliduj dane wejściowe

Stwórz formularz w Excelu do wprowadzania danych (np. oceny studentów) i użyj **walidacji danych** (listy rozwijane, zakresy wartości), aby upewnić się, że wprowadzane dane są poprawne.

Ćwiczenie 4: Oczyść dane z niepotrzebnych znaków

Wymyśl dane, które zawierają zbędne spacje, znaki specjalne lub powtarzające się fragmenty tekstu (np. " Produkt X “). Poćwicz ich **usuwanie** za pomocą funkcji tekstowych (np. `USUŃ.ZBĘDNE.SPACJE`) lub narzędzia”Znajdź i zamień".

Etap 7: Zaawansowane funkcje

Ćwiczenie 1: Oblicz sumę sprzedaży dla konkretnego produktu i regionu

Użyj funkcji **SUMA.WARUNKI** (lub SUMA.JEŻELI, jeśli masz tylko jeden warunek) dla danych sprzedażowych.

Ćwiczenie 2: Wylicz średnią ocenę filmów z określonego gatunku i reżysera

Zastosuj funkcję **ŚREDNIA.JEŻELI** lub **ŚREDNIA.WARUNKÓW**.

Ćwiczenie 3: Oblicz wiek osób na podstawie daty urodzenia

Użyj funkcji **DZIŚ/TERAZ** i operacji na datach, aby obliczyć, ile lat ma dana osoba i ile dni pozostało do jej urodzin.

Ćwiczenie 4: Znajdź maksymalną i minimalną temperaturę w danym miesiącu i mieście

Wykorzystaj funkcje **MAKS.WARUNKÓW** i **MIN.WARUNKÓW**.

Excel - quiz

Etap 1: Podstawy (niezbędne minimum)

Pytanie 1: Jakie klawisze lub kombinacje klawiszy służą do szybkiego poruszania się po arkuszu w Excelu?

Odpowiedź 1: Do poruszania się po arkuszu służą klawisze strzałek, a do szybszego przemieszczania się można używać kombinacji Ctrl+strzałki lub klawiszy Page Up/Down.

Pytanie 2: Wymień trzy podstawowe typy danych, które rozpoznaje Excel, oraz jeden typ oznaczający błąd.

Odpowiedź 2: Excel rozpoznaje podstawowe typy danych takie jak Tekst, Liczby oraz Daty. Jednym z typów oznaczających błąd jest np. #DIV/0!.

Pytanie 3: Czym różni się adresowanie względne od bezwzględnego w Excelu? Podaj przykład adresowania bezwzględnego.

Odpowiedź 3: Adresowanie względne (np. A1) zmienia się automatycznie przy kopiowaniu formuły, podczas gdy adresowanie bezwzględne (np. A1) pozostaje niezmienne, blokując zarówno kolumnę, jak i wiersz.

Etap 2: Formatowanie i organizacja danych

Pytanie 1: Podaj dwa przykłady, jak można sformatować komórki w Excelu, aby poprawić czytelność danych.

Odpowiedź 1: Można zmienić kolory tekstu i tła lub zastosować ramki i obramowania komórek.

Pytanie 2: Jakie dwie techniki organizacji danych pomagają zarządzać dużymi zbiorami danych w arkuszu?

Odpowiedź 2: Przydatne techniki to sortowanie danych według określonych kryteriów oraz filtrowanie danych (np. za pomocą autofiltru), aby wyświetlić tylko interesujące wiersze.

Etap 3: Podstawowe formuły (najważniejsze funkcje)

Pytanie 1: Jakiej funkcji użyjesz w Excelu, aby obliczyć średnią arytmetyczną z zakresu komórek?

Odpowiedź 1: Aby obliczyć średnią arytmetyczną, użyjesz funkcji ŚREDNIA.

Pytanie 2: Wymień dwie funkcje statystyczne służące do znalezienia skrajnych wartości w zbiorze danych.

Odpowiedź 2: Do znalezienia skrajnych wartości używa się funkcji MIN (dla wartości minimalnej) oraz MAX (dla wartości maksymalnej).

Etap 4: Funkcje logiczne, tekstowe i wyszukiwania (podstawy)

Pytanie 1: Do czego służy funkcja JEŻELI w Excelu?

Odpowiedź 1: Funkcja JEŻELI służy do implementacji podstawowej logiki warunkowej, wykonując różne działania w zależności od spełnienia określonego warunku.

Pytanie 2: Wymień dwie funkcje tekstowe służące do wyodrębnienia części tekstu z komórki.

Odpowiedź 2: Do wyodrębnienia części tekstu można użyć funkcji LEWY (z początku tekstu), PRAWY (z końca tekstu) lub FRAGMENT.TEKSTU (ze środka tekstu).

Pytanie 3: Jakie jest podstawowe zastosowanie funkcji WYSZUKAJ.PIONOWO (VLOOKUP)?

Odpowiedź 3: WYSZUKAJ.PIONOWO (VLOOKUP) służy do podstawowego wyszukiwania wartości w pierwszej kolumnie zakresu i zwracania wartości z innej kolumny w tym samym wierszu.

Etap 5: Wizualizacja danych

Pytanie 1: Jaki typ wykresu najlepiej nadaje się do przedstawienia trendów danych w czasie?

Odpowiedź 1: Do przedstawienia trendów w czasie najlepiej nadaje się wykres liniowy.

Pytanie 2: Wymień dwa podstawowe elementy każdego wykresu, które pomagają go zrozumieć.

Odpowiedź 2: Podstawowe elementy to m.in. tytuł wykresu, osie (X i Y) oraz legenda, jeśli występuje wiele serii danych.

Etap 6: Zaawansowane narzędzia

Pytanie 1: Do czego służą Tabele przestawne w Excelu?

Odpowiedź 1: Tabele przestawne służą do podsumowywania i analizowania dużych zbiorów danych, umożliwiając łatwe grupowanie danych i kalkulacje.

Pytanie 2: Wymień dwa rodzaje formatowania warunkowego, które wykraczają poza proste wyróżnianie komórek kolorem.

Odpowiedź 2: Zaawansowane rodzaje formatowania warunkowego to skale kolorów, paski danych oraz zestawy ikon.

Etap 7: Import i przetwarzanie danych, Zaawansowane funkcje

Pytanie 1: Z jakim potencjalnym problemem można się spotkać podczas wczytywania danych z plików CSV w Excelu?

Odpowiedź 1: Potencjalne problemy mogą dotyczyć separatorów (np. przecinek vs średnik) oraz kodowania znaków.

Pytanie 2: Podaj przykład funkcji agregującej z warunkami, której można użyć w Excelu.

Odpowiedź 2: Przykładem jest funkcja SUMA.JEŻELI lub SUMA.WARUNKI, które sumują wartości w zakresie spełniającym określone kryteria.

SQL - uniwersalny język danych

SQL (Structured Query Language) to język służący do zarządzania i przetwarzania danych w relacyjnych bazach danych. To najpotężniejszy język do analizy danych (serio!). Właściwie niezależnie od bazy danych (to uproszczenie, może nieco na wyrost) język SQL jest dostępny. Co więcej - w rozwiązaniach czysto programistycznych (np. w języku Python z biblioteką Pandas, albo w Sparku - narzędziu do operowania na bardzo dużych zbiorach danych) często da się użyć SQLa. Warto więc go znać, chociaż w podstawowym zakresie. Albo nie - nie warto. **Trzeba!**

Proponowany przeze mnie plan nauki SQL obejmuje:

- **Etap 1: Podstawy i środowisko pracy** (pojęcia baz danych, rodzaje systemów bazodanowych SQL, instalacja środowiska, narzędzia do pracy).
- **Etap 2: Podstawy SQL - Odczytywanie danych** (zapytania SELECT, filtrowanie danych za pomocą WHERE, operatory logiczne).
- **Etap 3: Funkcje i agregacje** (funkcje tekstowe, matematyczne, dat, funkcje agregujące jak COUNT, SUM, AVG, MIN, MAX, grupowanie danych za pomocą GROUP BY i HAVING).
- **Etap 4: Łączenie tabel (JOIN)** (rodzaje JOIN-ów: INNER, LEFT, RIGHT, FULL OUTER, CROSS, praktyka łączenia wielu tabel, aliasy).
- **Etap 5: Zaawansowane zapytania** (podzapytania, wyrażenia warunkowe CASE WHEN, funkcje okna).
- **Etap 6: Modyfikowanie danych** (dodawanie, aktualizowanie, usuwanie danych za pomocą INSERT, UPDATE, DELETE, TRUNCATE).
- **Etap 7: Struktura bazy danych** (tworzenie i modyfikowanie tabel CREATE TABLE, ALTER TABLE, DROP TABLE, ograniczenia (Constraints) jak PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL, INDEX-y).
- **Etapy 8 i 9** obejmują zaawansowane funkcje specyficzne dla PostgreSQL, optymalizację zapytań, transakcje, kopie zapasowe i migracje.

I tutaj nie polecam drogi na skróty, aby przeskoczyć do kolejnego rozdziału. O ile być może pierwsze etapy (powiedzmy do czwartego włącznie) są dość podstawowe, o tyle warto zobaczyć na kolejne.

SQL - plan nauki

Etap 1: Podstawy i środowisko pracy

Dlaczego? Bo to narzędzie (bazy danych), które trzeba poznać, wszystkie dane koniec końców *leżą w jakiejś bazie* (nawet jeśli to *data lake* a nie stricte baza danych). Trzeba poznać odmiany tego narzędzia, dodatki do tego narzędzia, zasady współpracy.

1. Czym jest SQL i bazy danych

- Relacyjne bazy danych - podstawowe pojęcia
- Tabele, kolumny, wiersze, klucze główne i obce
- Normalizacja baz danych (podstawy)
- SQL vs NoSQL - różnice

2. Rodzaje systemów bazodanowych SQL

- **PostgreSQL** - open source, bardzo funkcjonalny (główny focus)
- **MySQL/MariaDB** - popularny w aplikacjach webowych
- **SQLite** - lekka baza do małych projektów i nauki
- **Microsoft SQL Server** - enterprise, środowiska Microsoft
- **Oracle Database** - enterprise, bardzo zaawansowany
- **Różnice w składni** - podstawowe różnice między systemami

3. Instalacja i konfiguracja środowiska

- Instalacja PostgreSQL (lokalnie lub Docker)
- Tworzenie pierwszej bazy danych
- Tworzenie użytkownika i nadawanie uprawnień

4. Narzędzia do pracy z bazami danych

Narzędzia graficzne

- **DBeaver** - uniwersalny, darmowy, obsługuje wszystkie bazy
- **pgAdmin** - dedykowany dla PostgreSQL
- **HeidiSQL** - lekki, głównie MySQL/PostgreSQL
- **DataGrip** - płatny JetBrains, bardzo zaawansowany
- **Azure Data Studio** - Microsoft, cross-platform

Linia poleceń

- **psql** - oficjalny klient PostgreSQL
- Podstawowe komendy psql: `\l`, `\c`, `\dt`, `\d`, `\q`
- Wykonywanie skryptów z pliku
- Historia poleceń i edycja

Etap 2: Podstawy SQL - Odczytywanie danych

Dlaczego? Bo SQL to podstawa *rozmawiania* z danymi. Ma ponad 50 lat (!) i jest standardem. Nie znać SQL (nawet samych podstaw) w świecie danych to jak nie znać (precyzyjniej: nie umieć przeczytać) angielskiego w świecie IT.

5. Podstawowe zapytania SELECT

- **SELECT** - wybieranie kolumn
- **FROM** - określanie tabeli
- **DISTINCT** - unikalne wartości
- **LIMIT/TOP** - ograniczanie liczby wyników
- Komentarze w SQL (`--` i `/* */`)

6. Filtrowanie danych

- **WHERE** - warunki podstawowe
- Operatory porównania: `=`, `!=`, `<>`, `<`, `>`, `<=`, `>=`
- **IN** - sprawdzanie wielu wartości
- **BETWEEN** - zakresy wartości
- **LIKE** - wzorce tekstowe (`%`, `_`)
- **IS NULL / IS NOT NULL** - sprawdzanie wartości pustych

7. Operatory logiczne

- **AND** - wszystkie warunki muszą być spełnione
- **OR** - przynajmniej jeden warunek
- **NOT** - negacja warunku
- Grupowanie warunków nawiasami
- Kolejność wykonywania operatorów

8. Sortowanie danych

- **ORDER BY** - sortowanie wyników
- **ASC** (rosnąco) vs **DESC** (malejąco)

- Sortowanie po wielu kolumnach
- Sortowanie z NULL-ami

Etap 3: Funkcje i grupowanie

Dlaczego? Bo pojedyncza dana (jeden rekord) to często za mało. Bardziej interesuje nas łączna sprzedaż niż lista kolejnych elementów z paragonów z całego miesiąca w hipermarkecie.

9. Funkcje wbudowane

Funkcje tekstowe

- **CONCAT** - łączenie tekstów
- **UPPER/LOWER** - zmiana wielkości liter
- **LENGTH** - długość tekstu
- **SUBSTRING** - wyciąganie części tekstu
- **TRIM** - usuwanie spacji
- **REPLACE** - zamiana tekstu

Funkcje numeryczne

- **ROUND** - zaokrąglanie
- **ABS** - wartość bezwzględna
- **FLOOR/CEIL** - zaokrąglanie w dół/górę
- **MOD** - reszta z dzielenia

Funkcje dat

- **NOW/CURRENT_TIMESTAMP** - bieżąca data i czas
- **DATE_PART/EXTRACT** - wyciąganie części daty
- **AGE** - różnica dat (PostgreSQL)
- **DATE_TRUNC** - obcinanie daty do jednostki

10. Funkcje agregujące

- **COUNT** - zliczanie wierszy
- **SUM** - sumowanie wartości
- **AVG** - średnia arytmetyczna
- **MIN/MAX** - wartości minimalne/maksymalne
- **COUNT(DISTINCT)** - zliczanie unikalnych wartości

11. Grupowanie danych

- **GROUP BY** - grupowanie wyników
- **HAVING** - filtrowanie grup (vs WHERE)
- Reguły grupowania - wszystkie kolumny w SELECT
- Grupowanie po wielu kolumnach

Etap 4: Łączenie tabel (JOIN)

Dlaczego? Bo często dane są rozdzielone na wiele tabel, aby ich nie dublować. Jeśli jeden klient kupuje wiele produktów to nie musisz mieć przy każdym produkcie imienia, nazwiska i adresu domowego klienta, prawda? Możesz to sobie dokleić (*join*).

12. Rodzaje JOIN-ów

- **INNER JOIN** - tylko pasujące rekordy
- **LEFT JOIN** - wszystkie z lewej tabeli
- **RIGHT JOIN** - wszystkie z prawej tabeli
- **FULL OUTER JOIN** - wszystkie rekordy
- **CROSS JOIN** - iloczyn kartezjański

13. Praktyka JOIN-ów

- Łączenie dwóch tabel
- Łączenie wielu tabel
- Self JOIN - łączenie tabeli z samą sobą
- Aliasy tabel (AS)
- Warunki ON vs WHERE przy JOIN

Etap 5: Zaawansowane zapytania

Dlaczego? Bo to wygodniejsze. Bo to droga na skróty. Bo to daje o wiele więcej możliwości.

14. Podzapytania (Subqueries)

- Podzapytania w WHERE
- Podzapytania w SELECT
- Podzapytania w FROM
- **EXISTS/NOT EXISTS**
- **ANY/ALL** z podzapytaniami

15. Wyrażenia warunkowe

- **CASE WHEN** - logika warunkowa
- **COALESCE** - pierwsza niepusta wartość
- **NULLIF** - zamiana na NULL przy równości

16. Funkcje okna (Window Functions)

- **PARTITION BY** - podział na grupy
- **ROW_NUMBER()** - numerowanie wierszy
- **RANK()/DENSE_RANK()** - ranking
- **LAG/LEAD** - poprzednia/następna wartość
- **SUM/AVG OVER** - sumy/średnie kroczące

Etap 6: Modyfikowanie danych

Dlaczego? Bo dane są nie tylko czytane z bazy, ale też w niej aktualizowane i do niej dodawane.

17. Dodawanie danych

- **INSERT INTO** - dodawanie pojedynczych rekordów
- **INSERT INTO ... SELECT** - kopiowanie danych
- **INSERT INTO ... VALUES** - wiele rekordów naraz
- **ON CONFLICT** (PostgreSQL) - obsługa konfliktów

18. Aktualizowanie danych

- **UPDATE** - modyfikowanie istniejących rekordów
- **UPDATE ... FROM** - aktualizacja z JOIN
- Warunki WHERE w UPDATE
- Aktualizowanie wielu kolumn

19. Usuwanie danych

- **DELETE FROM** - usuwanie rekordów
- **TRUNCATE** - szybkie usuwanie wszystkich danych
- **DELETE ... USING** - usuwanie z JOIN
- Kaskadowe usuwanie

Etap 7: Struktura bazy danych

Dlaczego? Bo kiedyś samodzielnie przygotujesz strukturę bazy, tak aby Tobie było wygodnie (albo aby dało się zrobić taki czy inny projekt).

20. Tworzenie i modyfikowanie tabel

- **CREATE TABLE** - tworzenie nowych tabel
- Typy danych w PostgreSQL: TEXT, INTEGER, DECIMAL, DATE, TIMESTAMP, BOOLEAN
- **ALTER TABLE** - modyfikowanie struktury
- **DROP TABLE** - usuwanie tabel

21. Ograniczenia (Constraints)

- **PRIMARY KEY** - klucz główny
- **FOREIGN KEY** - klucz obcy
- **UNIQUE** - unikalne wartości
- **NOT NULL** - wymagane wartości
- **CHECK** - własne warunki
- **DEFAULT** - wartości domyślne

22. Indeksy

- **CREATE INDEX** - tworzenie indeksów
- Kiedy używać indeksów
- **EXPLAIN** - analiza planów wykonania
- Indeksy złożone

Etap 8: Zaawansowane funkcje PostgreSQL

Dlaczego? Bo warto stosować odpowiednie narzędzia do konkretnych celów.

23. Typy danych PostgreSQL

- **JSON/JSONB** - przechowywanie JSON
- **ARRAY** - tablice
- **UUID** - unikalne identyfikatory
- **ENUM** - typy wyliczeniowe

24. Zaawansowane funkcje

- **CTE (Common Table Expressions)** - WITH

- **Rekurencyjne CTE** - hierarchie danych
- **LATERAL JOIN** - zaawansowane łączenia
- **UPSERT** - INSERT ... ON CONFLICT

25. Funkcje i procedury

- **CREATE FUNCTION** - własne funkcje
- **DO blocks** - bloki kodu
- Podstawy PL/pgSQL
- Triggery (podstawy)

Etap 9: Optymalizacja i administracja

Dlaczego? Żeby szef nie zapytał dlaczego czeka na raport kilka godzin.

26. Wydajność zapytań

- **EXPLAIN ANALYZE** - szczegółowa analiza
- Czytanie planów wykonania
- Optymalizacja JOIN-ów
- Partycjonowanie tabel (podstawy)

27. Transakcje

- **BEGIN/COMMIT/ROLLBACK** - kontrola transakcji
- **SAVEPOINT** - punkty kontrolne
- Izolacja transakcji
- Deadlock-i i ich unikanie

28. Kopie zapasowe i migracje

- **pg_dump/pg_restore** - kopie zapasowe
- **COPY** - import/export danych CSV
- Migracje struktury bazy
- Wersjonowanie schematu

Praktyczne wskazówki do nauki

Ćwiczenia praktyczne

- **Zacznij od przykładowych baz** - Northwind, Sakila, Chinook
- **Używaj psql regularnie** - naucz się pracy z linii poleceń

- **Praktykuj na prawdziwych danych** - pobierz otwarte zestawy danych
- **Rozwiązuj problemy stopniowo** - od prostych do złożonych

Przydatne komendy psql

```
\l          - lista baz danych
\c dbname   - połączenie z bazą
\dt         - lista tabel
\d table    - struktura tabeli
\q          - wyjście
\i file.sql - wykonanie pliku
\timing on  - pomiar czasu wykonania
```

Najczęstsze błędy początkujących

- Brak GROUP BY przy funkcjach agregujących
- Mylenie WHERE z HAVING
- Nieprawidłowe JOIN-y (iloczyn kartezjański)
- Zapominanie o ORDER BY
- Problemy z NULL-ami

Zasoby do nauki

- **PostgreSQL Tutorial** - oficjalna dokumentacja
- **SQLBolt** - interaktywne ćwiczenia
- **W3Schools SQL** - podstawy z przykładami
- **Mode Analytics SQL Tutorial** - zaawansowane techniki
- **PostgreSQL Exercises** - praktyczne zadania

Kolejne kroki (po opanowaniu podstaw)

- Zaawansowane funkcje PostgreSQL (PostGIS, Full-text search)
- NoSQL (MongoDB, Redis) dla porównania
- ORM-y (SQLAlchemy, Django ORM)
- Data warehousing i analityka
- Administracja bazą danych

SQL - ćwiczenia

Etap 1: Podstawy i środowisko pracy

Ćwiczenie 1: Zainstaluj wybraną bazę danych

PostgreSQL lub **SQLite** są polecane do nauki.

Ćwiczenie 2: Zainstaluj narzędzie graficzne do pracy z bazami danych

Spróbuj **DBeaver** (uniwersalny) lub **pgAdmin** (dedykowany dla PostgreSQL). Połącz się ze swoją zainstalowaną bazą danych.

Ćwiczenie 3: Stwórz swoją pierwszą bazę danych i użytkownika

Korzystając z wybranego narzędzia, wykonaj podstawowe operacje konfiguracyjne, takie jak tworzenie nowej bazy danych i użytkownika z uprawnieniami.

Ćwiczenie 4: Zrozum pojęcia relacyjnych baz danych

Przejrzyj schematy przykładowych baz danych (np. Northwind, Sakila) i spróbuj zidentyfikować tabele, kolumny, wiersze, klucze główne i obce. Zastanów się, dlaczego stosuje się normalizację.

Etap 2: Podstawy SQL - Odczytywanie danych

Ćwiczenie 1: Pobierz wszystkie dane z wybranej tabeli

Użyj zapytania `SELECT * FROM nazwa_tabeli;`. Następnie spróbuj wybrać tylko konkretne kolumny, np. imiona i nazwiska z tabeli `customers`.

Ćwiczenie 2: Filtruj dane, aby znaleźć konkretne rekordy

Znajdź wszystkich klientów z miasta ‘Warszawa’ używając klauzuli **WHERE** i operatora równości (`=`). Następnie poszukaj klientów z listy miast (`IN`) lub w określonym zakresie wieku (`BETWEEN`).

Ćwiczenie 3: Posortuj wyniki zapytania

Posortuj klientów alfabetycznie według nazwiska, a następnie imienia. Zobacz, jak działa sortowanie rosnące (**ASC**) i malejące (**DESC**). Spróbuj posortować po kolumnie, która może zawierać `NULL`.

Ćwiczenie 4: Użyj operatorów logicznych

Znajdź klientów z ‘Warszawy’ lub ‘Krakowa’, którzy jednocześnie mają więcej niż 30 lat, stosując operatory **AND**, **OR**, **NOT** oraz grupowanie nawiasami.

Ćwiczenie 5: Znajdź dane za pomocą wzorców tekstowych

Wyszukaj wszystkie produkty, których nazwa zaczyna się na ‘A’ lub zawiera ‘lamp’, używając operatora **LIKE** (%, _).

Etap 3: Funkcje i agregacje

Ćwiczenie 1: Oblicz sumę sprzedaży i średnią cenę produktów

Wykorzystaj funkcje agregujące **SUM** i **AVG** dla danych sprzedażowych. Oblicz też **MIN** i **MAX** wartości.

Ćwiczenie 2: Zlicz, ilu jest klientów w każdym kraju i regionie

Użyj funkcji **COUNT** i klauzuli **GROUP BY**.

Ćwiczenie 3: Znajdź kraje, w których jest więcej niż 10 klientów

Zastosuj klauzulę **HAVING** po grupowaniu, aby filtrować grupy, a nie pojedyncze wiersze.

Ćwiczenie 4: Przekształć teksty i liczby

Zmień wszystkie nazwy produktów na wielkie litery za pomocą funkcji **UPPER**. Oblicz długość tekstów (`LENGTH`) lub zaokrąglaj wartości numeryczne (`ROUND`).

Ćwiczenie 5: Oblicz wiek klientów na podstawie daty urodzenia

Użyj funkcji datowych, takich jak **NOW/CURRENT_TIMESTAMP** i **DATE_PART/EXTRACT** (dla PostgreSQL **AGE**).

Etap 4: Łączenie tabel (JOIN)

Ćwiczenie 1: Połącz dane klientów z ich zamówieniami

Użyj **INNER JOIN**, aby zobaczyć tylko tych klientów, dla których istnieje pasujący rekord zamówienia.

Ćwiczenie 2: Wyświetl wszystkich klientów i ich zamówienia (jeśli takie są)

Zastosuj **LEFT JOIN**, aby zachować wszystkich klientów z lewej tabeli, nawet jeśli nie złożyli zamówień. Zobacz różnicę z **RIGHT JOIN**.

Ćwiczenie 3: Połącz trzy tabele

Jeśli masz tabelę produktów, kategorii i dostawców, spróbuj połączyć je wszystkie, aby wyświetlić nazwę produktu, jego kategorię i nazwę dostawcy.

Ćwiczenie 4: Użyj aliasów dla tabel

Zapisz zapytania z użyciem aliasów tabel (AS), aby kod był bardziej czytelny i zwięzły. Zastanów się, kiedy warunek `ON` jest lepszy od `WHERE` przy JOINach.

Ćwiczenie 5: Wykonaj Self JOIN

Znajdź wszystkich pracowników, którzy raportują do konkretnego menedżera, łącząc tabelę `employees` z samą sobą.

Etap 5: Zaawansowane zapytania

Ćwiczenie 1: Znajdź klientów, którzy złożyli więcej niż średnią liczbę zamówień

Użyj **podzapytania (Subquery)** w klauzuli **WHERE**. Spróbuj też użyć podzapytania w klauzuli `SELECT` do obliczenia jakiejś wartości agregującej dla każdego wiersza.

Ćwiczenie 2: Zastosuj logikę warunkową w zapytaniu

Utwórz nową kolumnę ‘StatusKlienta’, która na podstawie liczby zamówień lub wartości zakupów przypisuje ‘Nowy’, ‘Standardowy’ lub ‘Premium’ używając wyrażenia **CASE WHEN**.

Ćwiczenie 3: Użyj funkcji okna do rankingowania

Zrankuj klientów według sumy ich wydatków lub produktów, używając **ROW_NUMBER()** lub **RANK()** z **PARTITION BY**.

Ćwiczenie 4: Oblicz sumę kroczącą sprzedaży

Wylicz sumę sprzedaży dla każdego dnia i wszystkich poprzednich dni (lub ostatnie 7 dni), używając funkcji okna **SUM OVER()**.

Ćwiczenie 5: Wykorzystaj COALESCE i NULLIF

Użyj `COALESCE` do zwrócenia pierwszej niepustej wartości z listy kolumn (np. numer telefonu służbowy, a jeśli brak, to prywatny). Użyj `NULLIF` do zamiany pustego ciągu znaków na NULL.

Etap 6: Modyfikowanie danych

Ćwiczenie 1: Dodaj nowego klienta do tabeli

Użyj komendy **INSERT INTO** z konkretnymi wartościami. Spróbuj też dodać wiele rekordów naraz.

Ćwiczenie 2: Zaktualizuj dane istniejącego klienta

Zmień adres e-mail klienta o konkretnym ID za pomocą komendy **UPDATE**. Spróbuj zaktualizować wiele kolumn jednocześnie.

Ćwiczenie 3: Usuń rekordy, które spełniają warunek

Usuń wszystkich klientów, którzy nie złożyli żadnego zamówienia od roku.

Ćwiczenie 4: Szybko wyczyść tabelę

Wykorzystaj **TRUNCATE** na tabeli testowej, aby zobaczyć, jak szybko usuwa wszystkie dane, w porównaniu do `DELETE FROM`.

Etap 7: Struktura bazy danych

Ćwiczenie 1: Stwórz nową tabelę ‘produkty’ z kolumnami ID, nazwa, cena, data_dodania

Zdefiniuj odpowiednie typy danych (np. `INTEGER`, `TEXT`, `DECIMAL`, `DATE`) dla każdej kolumny.

Ćwiczenie 2: Zmodyfikuj istniejącą tabelę

Dodaj nową kolumnę ‘opis’ do tabeli ‘produkty’. Zmień typ danych istniejącej kolumny.

Ćwiczenie 3: Dodaj ograniczenia do tabeli

Zdefiniuj **PRIMARY KEY** dla ID produktu, **NOT NULL** dla nazwy produktu i **UNIQUE** dla kolumny SKU (fikcyjny kod produktu).

Ćwiczenie 4: Utwórz klucz obcy

Zdefiniuj **FOREIGN KEY** łączący tabelę zamówień z tabelą klientów, aby zapewnić integralność danych.

Ćwiczenie 5: Stwórz indeks

Stwórz indeks na kolumnie `nazwa_produktu`, a następnie użyj `EXPLAIN` do analizy planu wykonania zapytania, aby zobaczyć, jak indeks wpływa na wydajność zapytań.

Etap 8: Zaawansowane funkcje PostgreSQL

Ćwiczenie 1: Użyj JSONB do przechowywania danych

Stwórz tabelę z kolumną JSONB i przechowuj w niej dane o konfiguracji produktu, a następnie spróbuj odpytać dane z wnętrza struktury JSON.

Ćwiczenie 2: Zastosuj CTE (Common Table Expressions) do złożonych zapytań

Napisz zapytanie, które oblicza średnią sprzedaż w danym miesiącu, a następnie używa tego wyniku w innym CTE do dalszych obliczeń.

Ćwiczenie 3: Zdefiniuj prostą funkcję bazodanową

Napisz funkcję, która wylicza podatek od ceny produktu lub formatuje adres, używając podstaw PL/pgSQL.

Ćwiczenie 4: Wykorzystaj rekurencyjne CTE

Zbuduj zapytanie, które pobiera strukturę hierarchiczną (np. drzewo kategorii produktów lub strukturę organizacyjną) za pomocą rekurencyjnego CTE.

Etap 9: Optymalizacja i administracja

Ćwiczenie 1: Zanalizuj plan wykonania zapytania

Użyj **EXPLAIN ANALYZE** dla złożonego zapytania z `JOIN`ami i spróbuj zidentyfikować, gdzie baza danych spędza najwięcej czasu (wąskie gardła).

Ćwiczenie 2: Poćwicz transakcje

Otwórz transakcję (**BEGIN**), wykonaj kilka operacji `INSERT` lub `UPDATE`, a następnie **ROLLBACK**, aby cofnąć zmiany. Następnie spróbuj z **COMMIT**, aby trwale zapisać zmiany.

Ćwiczenie 3: Stwórz kopię zapasową bazy danych

Użyj narzędzi takich jak `pg_dump` (dla PostgreSQL) do utworzenia kopii zapasowej swojej testowej bazy danych.

Ćwiczenie 4: Eksportuj i importuj dane CSV za pomocą `COPY`

Poćwicz szybki eksport danych z tabeli do pliku CSV i import z powrotem do innej tabeli.

SQL - quiz

Etap 1: Podstawy i środowisko pracy

Pytanie 1: Czym jest SQL w kontekście pracy z bazami danych?

Odpowiedź 1: SQL (Structured Query Language) jest językiem niezbędnym do pracy z bazami danych, umożliwiającym przetwarzanie dużych zbiorów informacji.

Pytanie 2: W relacyjnych bazach danych, czym różni się tabela od wiersza i kolumny?

Odpowiedź 2: Tabela to główna struktura przechowująca dane. Wiersz reprezentuje pojedynczy rekord danych, a kolumna reprezentuje konkretny atrybut lub typ informacji w tabeli.

Pytanie 3: Wymień dwa przykładowe graficzne narzędzia do pracy z bazami danych.

Odpowiedź 3: Przykładami graficznych narzędzi są DBeaver (uniwersalny) lub pgAdmin (dedykowany dla PostgreSQL).

Etap 2: Podstawy SQL - Odczytywanie danych

Pytanie 1: Jaką klauzulę w zapytaniu SELECT użyjesz do wskazania tabeli, z której chcesz pobrać dane?

Odpowiedź 1: Do wskazania tabeli użyjesz klauzuli FROM.

Pytanie 2: Jakie jest zastosowanie klauzuli WHERE w zapytaniu SELECT?

Odpowiedź 2: Klauzula WHERE służy do filtrowania danych, określając warunki, które muszą spełniać wiersze, aby zostały zwrócone.

Pytanie 3: Jak uporządkujesz wyniki zapytania SQL według rosnącej wartości konkretnej kolumny?

Odpowiedź 3: Do uporządkowania wyników użyjesz klauzuli ORDER BY wraz ze słowem kluczowym ASC.

Etap 3: Funkcje i grupowanie

Pytanie 1: Jakiej funkcji wbudowanej w SQL użyjesz, aby połączyć kilka tekstów w jeden?

Odpowiedź 1: Do łączenia tekstów użyjesz funkcji CONCAT.

Pytanie 2: Wymień trzy przykładowe funkcje agregujące w SQL, które podsumowują dane w grupach.

Odpowiedź 2: Przykładami funkcji agregujących są COUNT (zliczanie), SUM (sumowanie) oraz AVG (średnia arytmetyczna).

Pytanie 3: Jaka klauzula służy do filtrowania wyników po ich pogrupowaniu za pomocą GROUP BY?

Odpowiedź 3: Do filtrowania grup używa się klauzuli HAVING, w przeciwieństwie do WHERE, która filtruje wiersze przed grupowaniem.

Etap 4: Łączenie tabel (JOIN)

Pytanie 1: Czym charakteryzuje się INNER JOIN?

Odpowiedź 1: INNER JOIN zwraca tylko te wiersze, dla których istnieje pasujący rekord w obu łączonych tabelach.

Pytanie 2: Jaki typ JOIN zwraca wszystkie wiersze z lewej tabeli oraz pasujące wiersze z prawej tabeli?

Odpowiedź 2: Ten opis odpowiada typowi LEFT JOIN.

Etap 5: Zaawansowane zapytania

Pytanie 1: Co to jest podzapytanie (subquery) w SQL i gdzie może być użyte w zapytaniu SELECT?

Odpowiedź 1: Podzapytanie to zapytanie zagnieżdżone w innym zapytaniu. Może być użyte m.in. w klauzulach WHERE, SELECT lub FROM.

Pytanie 2: Do czego służy wyrażenie CASE WHEN?

Odpowiedź 2: CASE WHEN pozwala na implementację logiki warunkowej w zapytaniu SQL, przypisując różne wartości w zależności od spełnienia określonych warunków.

Pytanie 3: Wymień jedną przykładową funkcję okna (Window Function) i opisz jej zastosowanie.

Odpowiedź 3: Przykładem jest funkcja ROW_NUMBER(), która numeruje wiersze w partycji wynikowej zapytania. Inne to RANK() (ranking) czy SUM/AVG OVER (sumy/średnie kroczące).

Etap 6: Modyfikowanie danych

Pytanie 1: Jakiej komendy SQL użyjesz, aby dodać nowy wiersz do tabeli?

Odpowiedź 1: Do dodania nowego wiersza użyjesz komendy INSERT INTO.

Pytanie 2: Jak zmodyfikujesz istniejące rekordy w tabeli?

Odpowiedź 2: Do modyfikacji istniejących rekordów służy komenda UPDATE.

Pytanie 3: Czym różni się komenda DELETE FROM od TRUNCATE przy usuwaniu danych z tabeli?

Odpowiedź 3: DELETE FROM usuwa wiersze pojedynczo (z możliwością użycia WHERE i wyzwalaczy), podczas gdy TRUNCATE jest szybszą komendą, która usuwa wszystkie dane z tabeli jednocześnie.

Etap 7: Struktura bazy danych

Pytanie 1: Jaką komendę SQL użyjesz, aby stworzyć nową tabelę w bazie danych?

Odpowiedź 1: Do stworzenia nowej tabeli użyjesz komendy CREATE TABLE.

Pytanie 2: Wymień dwa przykładowe ograniczenia (constraints), które można zdefiniować dla kolumn w tabeli, aby zapewnić integralność danych.

Odpowiedź 2: Przykładami są PRIMARY KEY (klucz główny), FOREIGN KEY (klucz obcy), UNIQUE (unikalne wartości) lub NOT NULL (wymagane wartości).

Etap 8: Zaawansowane funkcje PostgreSQL

Pytanie 1: Wymień dwa przykładowe zaawansowane typy danych dostępne w PostgreSQL.

Odpowiedź 1: Przykłady to JSON/JSONB (przechowywanie JSON), ARRAY (tablice), UUID czy ENUM.

Pytanie 2: Do czego służy CTE (Common Table Expression), definiowane za pomocą słowa kluczowego WITH?

Odpowiedź 2: CTE (WITH) pozwala na zdefiniowanie tymczasowego, nazwanego zbioru wyników, który można odwoływać w ramach pojedynczego zapytania, co poprawia czytelność i modularność.

Etap 9: Optymalizacja i administracja

Pytanie 1: Jakiej komendy w SQL użyjesz, aby analizować plan wykonania zapytania i zrozumieć, jak baza danych przetwarza dane?

Odpowiedź 1: Do analizy planów wykonania zapytania używa się komendy EXPLAIN, a EXPLAIN ANALYZE dostarcza szczegółowej analizy z czasami wykonania.

Pytanie 2: W kontekście transakcji w SQL, jakie są podstawowe komendy do kontrolowania ich przebiegu?

Odpowiedź 2: Podstawowe komendy to BEGIN (rozpoczęcie transakcji), COMMIT (zapisanie zmian) oraz ROLLBACK (cofnięcie zmian).

Statystyka - czasem trzeba znać teorię

Dlaczego statystyka i matematyka są ważne dla analityka danych i jakie pojęcia należy znać?

Statystyka i matematyka stanowią solidny fundament analizy danych i są niezbędne do wyciągania znaczących wniosków z dużych zbiorów danych. Analityk danych powinien znać podstawowe pojęcia (i umieć je wyjaśnić komuś, kto ich nie zna!) takie jak:

- **Miary tendencji centralnej** (średnia, mediana, moda).
- **Miary rozproszenia** (odchylenie standardowe, wariancja).
- Prawdopodobieństwo i rozkłady prawdopodobieństwa.
- Korelacja i regresja.
- Testowanie hipotez.

Zrozumienie tych pojęć pozwala na walidację jakości danych, monitorowanie procesów, rzetelną interpretację wyników, raportowanie z uwzględnieniem przedziałów ufności oraz efektywną komunikację wyników z biznesem (dlatego należy umieć wyjaśnić te pojęcia - nie każdy jest analitykiem danych, a jako analityk danych chcemy przedstawić efekty naszej pracy, tak aby były zrozumiałe, prawda?).

Do czego statystyka przydaje się w ramach odpowiednich specjalności?

Data Analyst/Raportowanie/BI:

- Rzetelna interpretacja wyników
- Dobór odpowiednich metryk (KPI) - lepiej średnia czy mediana?
- Porównania między okresami/grupami, testy hipotez
- Komunikacja niepewności

Data Engineer:

- Walidacja jakości danych (outliers, rozkłady)
- Monitoring pipeline'ów (anomalie, trendy)
- A/B testing infrastruktury

Poziom, który tutaj proponuję poznać jest wystarczający do profesjonalnej pracy, ale **nie dotyczy to zaawansowanego uczenia maszynowego czy sztucznej inteligencji** (ML/AI) - tu przydałoby się więcej. Aby profesjonalnie zajmować się uczeniem maszynowym, potrzebujesz rozszerzyć swoją wiedzę statystyczną w kilku kluczowych obszarach:

- Statystyka bayesowska

  - Twierdzenie Bayesa i jego zastosowania w ML
  - Założenia a priori i a posteriori
  - Bayesowskie sieci neuronowe i wnioskowanie bayesowskie
  - Metody Monte Carlo i MCMC
- Zaawansowane rozkłady prawdopodobieństwa
  - Rozkłady wykładnicze, gamma, beta
  - Rozkłady wielowymiarowe (wielowymiarowy normalny)
  - Mieszanki rozkładów (Gaussian Mixture Models)
  - Rozkłady warunkowe
- Algebra liniowa dla ML
  - Normy wektorowe i macierzowe
  - Własności macierzy (dodatnio określone, półokreślone)
  - Rozkład wartości osobliwych (SVD)
  - Rzutowanie ortogonalne i przestrzenie liniowe
- Optymalizacja matematyczna
  - Metody gradientowe (gradient descent, stochastic GD)
  - Regularyzacja (L1, L2, elastic net)
  - Wypukłość funkcji i optimum globalne vs lokalne
  - Mnożniki Lagrange’a i optymalizacja z ograniczeniami
- Statystyka wielowymiarowa
  - Miary podobieństwa i odległości
  - Analiza składowych głównych (PCA)
- Teoria próbkowania i cross-validation

Statystyka - ćwiczenia

Etap 1: Miary tendencji centralnej i rozproszenia

Ćwiczenie 1: Oblicz średnią, medianę i modę dla różnych zbiorów danych

Zbierz dane o zarobkach (np. fikcyjnych 20 osób) i wylicz dla nich te miary. **Porównaj wyniki** i zastanów się, która miara najlepiej opisuje "typowy" zarobek w zależności od występowania wartości odstających.

Ćwiczenie 2: Zrozum odchylenie standardowe i wariancję

Wylicz odchylenie standardowe dla danych o ocenach z egzaminu dla dwóch różnych grup studentów (jednej z bardzo zbliżonymi wynikami, drugiej z dużym rozstrzałem). Zastanów się, co większe odchylenie oznacza w praktyce.

Ćwiczenie 3: Wykryj outliery (wartości odstające)

Wymyśl dane o wzroście ludzi (np. 10 wartości), dodając jedną ekstremalnie wysoką lub niską wartość (np. wzrost 3 metry). Zobacz, jak **średnia** reaguje na takie odstające wartości, a jak **mediana**.

Ćwiczenie 4: Zastosuj KPI

Dla fikcyjnych danych sprzedażowych firmy (np. wartość pojedynczych transakcji) zdecyduj, czy lepszym **KPI** do monitorowania jest średnia wartość transakcji, czy mediana. Uzasadnij swój wybór.

Etap 2: Prawdopodobieństwo, korelacja i regresja

Ćwiczenie 1: Analizuj korelację

Zbierz dane o temperaturze i sprzedaży lodów w ciągu miesiąca. Oblicz **współczynnik korelacji** (np. Pearsona) i zastanów się, czy wysoka korelacja oznacza **przyczynowość**. Wymyśl sytuację, gdzie korelacja jest wysoka, ale przyczynowości brak (np. sprzedaż lodów i liczba utonięć).

Ćwiczenie 2: Stwórz prosty model regresji liniowej

Na podstawie fikcyjnych danych o latach doświadczenia i zarobkach, stwórz prosty **model regresji liniowej** (ręcznie lub w Pythonie), aby przewidzieć zarobki. Zastanów się, co oznaczają współczynniki modelu.

Ćwiczenie 3: Zrozum rozkłady prawdopodobieństwa

Wyobraź sobie wyniki rzutu kostką (rozkład jednostajny) i wyniki egzaminu dla dużej grupy studentów (rozkład normalny). Zastanów się, czym się różnią i kiedy który rozkład jest bardziej prawdopodobny.

Ćwiczenie 4: Oblicz prawdopodobieństwo

Jeśli masz 10 czerwonych i 5 niebieskich kulek w urnie, oblicz prawdopodobieństwo wylosowania niebieskiej kulki. Następnie oblicz prawdopodobieństwo wylosowania dwóch czerwonych kulek bez zwracania.

Etap 3: Testowanie hipotez

Ćwiczenie 1: Zaprojektuj prosty test A/B

Wymyśl scenariusz dla sklepu internetowego (np. dwie wersje przycisku “Kup teraz”). Zastanów się, jaką **hipotezę zerową i alternatywną** postawisz, aby sprawdzić, czy jedna wersja przycisku jest lepsza od drugiej. Określ, jakie dane byś zbierał.

Ćwiczenie 2: Zinterpretuj p-value

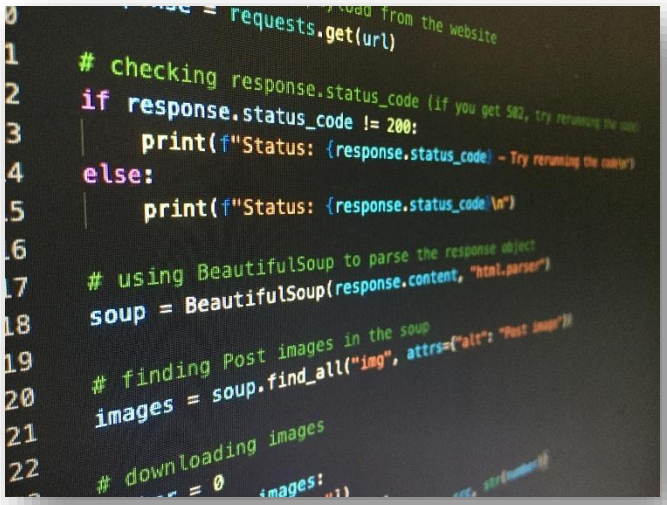
Znajdź w internecie przykłady wyników testów statystycznych (np. t-testu) z wartością p-value i spróbuj wyjaśnić, co oznacza wynik (np. $p<0.05$ vs $p>0.05$) w kontekście odrzucenia lub nieodrzucenia hipotezy zerowej.

Ćwiczenie 3: Porównaj grupy

Mając fikcyjne dane o średniej sprzedaży z dwóch różnych kampanii marketingowych, zastanów się, jak statystycznie sprawdzić (bez wykonywania obliczeń), czy różnica w sprzedaży jest statystycznie istotna. Opisz kroki testowania hipotez w tym kontekście.

Python - język (programowania) danych

Następnym etapem jest opanowanie podstaw programowania, szczególnie w języku Python, który jest szeroko stosowany w analizie danych. Według raportu Stack Overflow, Python i SQL są jednymi z najpopularniejszych języków programowania, używanymi przez ponad 49% respondentów.



Tutaj uwaga - jeśli chcesz być tylko człowiekiem od robienia raportów w jakimś BI narzędziu (PowerBI, Tableau, Looker, Qlik, SAS Business Intelligence , SAS Visual Analytics) to właściwie **znajomość Pythona nie będzie Ci potrzebna**. Ale Python to język przyjazny dla początkujących, a jednocześnie niezwykle potężny w kontekście analizy danych. Co więcej - ostatnio Microsoft wprowadza Pythona gdzie może (od dawna jest w PowerBI, ostatnio nawet w Excelu), więc warto poznać ten język.

Uwaga kolejna - w tej książce nie wspominam o **języku R** mimo, że pisałem w nim kilka lat (są na to dowody na blogu). Osobiście uważam, że zaczynając naukę programowania *pod dane* szkoda na niego czasu. Koniec końców - przy modelach związanych z sieciami neuronowymi (PyTorch, Keras, TensorFlow) - skończysz z Pythonem. Idąc w stronę inżynierii danych - bez Pythona ani rusz, jeśli chcesz *ogarniać wszystkie aspekty*. Jednakowoż język R jest popularny w niektórych sektorach (farmacja; wiem że w okolicach polskiego Ministerstwa Zdrowia i przyległych instytucji z R korzysta się mocno).

Wracając do nauki Pythona - zacznij od podstawowej składni, struktur danych i kontroli przepływu. Z czasem przejdź do bibliotek specjalizowanych w analizie danych, takich jak Pandas, NumPy czy Matplotlib, które umożliwiają zaawansowaną manipulację danymi i ich wizualizację.

Kroki zalecane przeze mnie w nauce Pythona od podstaw, a w kierunku pracy z danymi to na początek trzy etapy:

- **Etap 1: Środowisko pracy i podstawy** (instalacja, narzędzia - IDE, edytory, Jupyter Notebook, podstawowe operacje).
- **Etap 2: Podstawy języka Python** (zmienne, typy danych, struktury danych - listy, krotki, słowniki, zbiory, kontrola przepływu - instrukcje warunkowe i pętle, funkcje).
- **Etap 3: Praca z danymi - biblioteki podstawowe** (praca z plikami, obsługa błędów, moduły i pakiety).

Dalsze etapy skupiają się na kluczowych bibliotekach do analizy danych: NumPy (obliczenia numeryczne), Pandas (manipulacja danych), Matplotlib i Seaborn (wizualizacja danych).

Python - plan nauki

Etap 1: Środowisko pracy i podstawy

Dlaczego? Bo aby programować w Pythonie trzeba tego Pythona *mieć*. A środowiska wirtualne to bardzo przydatna sprawa!

1. Instalacja i konfiguracja środowiska

- **Instalacja Pythona** - Python.org vs Anaconda Distribution
- **Anaconda/Miniconda** - zarządzanie pakietami i środowiskami
- **Conda vs pip** - instalowanie bibliotek
- **Środowiska wirtualne** - izolacja projektów
- W 2025 roku warto też poznać **uv** jako narzędzie świetnie zastępujące Anacondę czy też pip.

2. Narzędzia do programowania

IDE i edytory

- **Jupyter Notebook/JupyterLab** - idealne do analizy danych i eksploracji
- **Google Colab** - darmowe środowisko w chmurze
- **VS Code** - uniwersalny edytor z rozszerzeniami Python
- **PyCharm** - profesjonalne IDE (Community/Professional)
- **Spyder** - IDE dedykowane analizie danych, szczerze mówiąc nie znam nikogo kto używa tego edytora

Narzędzia pomocnicze

- **IPython** - interaktywna konsola Python
- **Git** - kontrola wersji (poznaj podstawy, ale bez gita ani rusz w pracy w zespole)
- **Terminal/Command Prompt** - podstawowe komendy

3. Pierwszy kontakt z Pythonem

- **Python jako kalkulator** - podstawowe operacje
- **print()** - wypisywanie na ekran
- **Komentarze** - # i """ """
- **Uruchamianie skryptów** - **.**py vs notebook i jego *celki*
- **Debugging** - podstawowe techniki debugowania

Etap 2: Podstawy języka Python

Dlaczego? Bo jakoś trzeba zacząć :)

4. Zmienne i typy danych

Podstawowe typy

- **int** - liczby całkowite
- **float** - liczby zmiennoprzecinkowe
- **str** - tekst/stringi
- **bool** - wartości logiczne (True/False)
- **None** - wartość pusta

Operacje na typach

- Operatory arytmetyczne: `+`, `-`, `*`, `/`, `//`, `%`, `**`
- Operatory porównania: `==`, `!=`, `<`, `>`, `<=`, `>=`
- Operatory logiczne: `and`, `or`, `not`
- Konwersja typów: `int()`, `float()`, `str()`, `bool()`

5. Struktury danych

Listy (list)

- Tworzenie list: `[1, 2, 3]`
- Indeksowanie i slicing: `lista[0]`, `lista[1:3]`
- Metody list: `append()`, `extend()`, `insert()`, `remove()`, `pop()`
- List comprehensions: `[x**2 for x in range(10)]`

Krotki (tuple)

- Niezmienne sekwencje: `(1, 2, 3)`
- Unpacking: `a, b, c = (1, 2, 3)`

Słowniki (dict)

- Pary klucz-wartość: `{'klucz': 'wartość'}`
- Dostęp do wartości: `slownik['klucz']`
- Metody: `keys()`, `values()`, `items()`, `get()`
- Dictionary comprehensions

Zbiory (set)

- Unikalne elementy: `{1, 2, 3}`

- Operacje na zbiorach: przecięcie, suma, różnica

6. Kontrola przepływu

Instrukcje warunkowe

- **if**, **elif**, **else**
- Operatory `in`, `not in`
- Zagnieżdżone warunki

Pętle

- **for** - iterowanie po sekwencjach
- **while** - pętle z warunkiem
- **range()** - generowanie sekwencji liczb
- **break** i **continue** - kontrola pętli
- **enumerate()** i **zip()** - zaawansowane iterowanie

7. Funkcje

- **def** - definiowanie funkcji
- Parametry i argumenty
- **return** - zwracanie wartości
- Argumenty domyślne
- `*args` i `**kwargs`
- **lambda** - funkcje anonimowe
- Funkcje wbudowane: `len()`, `sum()`, `min()`, `max()`, `sorted()`

Etap 3: Praca z danymi - biblioteki podstawowe

Dlaczego? Bo wbudowane w Pythona elementy często *robią robotę* bez użycia dodatkowych bibliotek. I robią to czasem szybciej!

8. Praca z plikami

- **open()** - otwieranie plików
- Tryby: `'r'`, `'w'`, `'a'`
- Encoding - kodowanie znaków narodowych (we współpracy z CSV z Excela to może się objawić)
- **with** - context manager
- Czytanie: `read()`, `readline()`, `readlines()`

- Zapisywanie: `write()`, `writelines()`
- Formaty: TXT, CSV, JSON

9. Obsługa błędów

- **try**, **except**, **finally**
- Rodzaje wyjątków: `ValueError`, `TypeError`, `FileNotFoundError`
- **raise** - rzucanie własnych wyjątków
- Debugging z `print()` i `pdb`

10. Moduły i pakiety

- **import** - importowanie modułów
- **from ... import** - selektywny import
- **as** - aliasy
- Moduły standardowej biblioteki:
  - **os** - operacje systemowe
  - **datetime** - praca z datami
  - **math** - funkcje matematyczne
  - **random** - liczby losowe
  - **json** - praca z JSON
  - **csv** - praca z CSV

Etap 4: NumPy - obliczenia numeryczne

Dlaczego? Bo NumPy działa na tablicach czy też macierzach, a dane tak z reguły są ułożone.

11. Podstawy NumPy

- **Instalacja**: `pip install numpy`
- **Import**: `import numpy as np`
- **Arrays vs listy** - różnice w wydajności
- Tworzenie arrays: `np.array()`, `np.zeros()`, `np.ones()`, `np.arange()`

12. Operacje na arrays

- **Indeksowanie i slicing** - podobnie jak listy, ale wielowymiarowe
- **Kształt arrays**: `shape`, `reshape()`, `flatten()`
- **Operacje matematyczne** - wektoryzacja
- **Broadcasting** - operacje na arrays różnych rozmiarów
- **Boolean indexing** - filtrowanie danych

13. Funkcje NumPy dla analizy danych

- **Statystyki**: `mean()`, `median()`, `std()`, `var()`, `min()`, `max()`
- **Agregacje**: `sum()`, `prod()`, `cumsum()`
- **Operacje na osiach** - parameter *axis*
- **Funkcje matematyczne**: `sqrt()`, `exp()`, `log()`, `sin()`, `cos()`

Etap 5: Pandas - manipulacja danymi

Dlaczego? Bo Pandas to podstawa w przetwarzaniu danych w Pythonie, tak jak SQL przy bazach danych.

14. Podstawy Pandas

- **Instalacja**: `pip install pandas`
- **Import**: `import pandas as pd`
- **Series** - jednowymiarowe dane z indeksem
- **DataFrame** - dwuwymiarowa tabela danych
- Tworzenie DataFrame z różnych źródeł - list, słowników, list list

15. Wczytywanie i zapisywanie danych

- **CSV**: `pd.read_csv()`, `df.to_csv()`
- **Excel**: `pd.read_excel()`, `df.to_excel()`
- **JSON**: `pd.read_json()`, `df.to_json()`
- **SQL**: `pd.read_sql()`, `df.to_sql()`
- **Parametry wczytywania** - separator, encoding, headers, usecols, dtype

16. Eksploracja danych

- **Podstawowe informacje**: `head()`, `tail()`, `info()`, `describe()`, `shape`
- **Typy danych**: `dtypes`, `astype()`
- **Unikalne wartości**: `unique()`, `value_counts()`, `nunique()`
- **Wartości brakujące**: `isnull()`, `notnull()`, `isna()`, `notna()`

17. Indeksowanie i selekcja

- **Kolumny**: `df['kolumna']`, `df[['kol1', 'kol2']]`
- **Wiersze**: `df.loc[]`, `df.iloc[]`
- **Filtrowanie**
- **query()** - SQL-like filtrowanie (wspominałem, że SQL warto znać?)

18. Czyszczenie i transformacja danych

- **Brakujące dane**: `dropna()`, `fillna()`, `interpolate()`
- **Duplikaty**: `duplicated()`, `drop_duplicates()`
- **Zamiana wartości**: `replace()`, `map()`
- **Stosowanie funkcji**: `apply()`, `applymap()`
- **Zmiana typów**: `astype()`, `pd.to_datetime()`, `pd.to_numeric()`

19. Grupowanie i agregacja

- **groupby()**, **aggregate()** - grupowanie danych
- **Funkcje agregujące**: `sum()`, `mean()`, `count()`, `agg()`
- **Grupowanie po wielu kolumnach**
- **transform()** vs **apply()** w grupach
- **Tabele przestawne**: `pivot_table()`, `crosstab()`

20. Łączenie danych

- **concat()** - łączenie DataFrame'ów
- **merge()** - łączenie jak SQL JOIN
- **join()** - łączenie po indeksie
- **Typy join-ów**: inner, outer, left, right

Etap 6: Matplotlib - wizualizacja danych

Dlaczego? Bo jeden obraz wart… tak - o wykresy chodzi.

21. Podstawy Matplotlib

- **Instalacja**: `pip install matplotlib`
- **Import**: `import matplotlib.pyplot as plt`
- **Podstawowy workflow**: figure, axes, plot, show
- **Jupyter notebook**: `%matplotlib inline`

22. Podstawowe typy wykresów

- **Wykres liniowy**: `plt.plot()`
- **Wykres punktowy**: `plt.scatter()`
- **Histogram**: `plt.hist()`
- **Wykres słupkowy**: `plt.bar()`, `plt.barh()`

- **Wykres kołowy**: `plt.pie()`
- **Boxplot**: `plt.boxplot()`

23. Formatowanie wykresów

- **Tytuły i etykiety**: `title()`, `xlabel()`, `ylabel()`
- **Legenda**: `legend()`
- **Siatka**: `grid()`
- **Kolory i style**: color, linestyle, marker
- **Figsize**: rozmiar wykresów
- **Subplots**: wiele wykresów na jednej figurze

24. Integracja Pandas z Matplotlib

- **df.plot()** - bezpośrednie plotowanie z DataFrame
- **Różne typy**: `kind='line'`, `kind='bar'`, `kind='hist'`
- **Grupowane wykresy** z groupby

Etap 7: Seaborn - zaawansowana wizualizacja

Dlaczego? Bo z Seaborn pisze się szybciej niż z Matplotlib.

25. Podstawy Seaborn

- **Instalacja**: `pip install seaborn`
- **Import**: `import seaborn as sns`
- **Style**: `sns.set_style()`, `sns.set_palette()`
- **Integracja z Pandas DataFrame**

26. Wykresy statystyczne

- **Rozkłady**: `distplot()`, `histplot()`, `kdeplot()`
- **Relacje**: `scatterplot()`, `lineplot()`, `regplot()`
- **Kategorie**: `barplot()`, `countplot()`, `boxplot()`, `violinplot()`
- **Macierze**: `heatmap()`, `clustermap()`

27. Wykresy wielowymiarowe

- **FacetGrid** - wykresy panelowe
- **pairplot()** - macierz wykresów
- **jointplot()** - wykresy z rozkładami marginalnymi

Etap 8: Projekty praktyczne i zaawansowane techniki

Dlaczego? Bo tylko projekty dają doświadczenie.

28. Pierwszy projekt analizy danych

- **Wybranie zestawu danych** (Kaggle, UCI ML Repository)
- **Eksploracyjna analiza danych (EDA)**:
  - Wczytanie i pierwsze spojrzenie na dane
  - Sprawdzenie jakości danych
  - Statystyki opisowe
  - Wizualizacje rozkładów
  - Korelacje między zmiennymi
  - Wnioski i insights

29. Zaawansowane techniki Pandas

- **MultiIndex** - hierarchiczne indeksy
- **Kategoryczne dane**: `pd.Categorical`
- **Szeregi czasowe**: `pd.date_range()`, resampling
- **Window functions**: `rolling()`, `expanding()`

30. Statystyka dla analizy danych

Miary tendencji centralnej

- **Średnia arytmetyczna** - `np.mean()`, `df.mean()`
- **Mediana** - `np.median()`, `df.median()`
- **Moda** - `scipy.stats.mode()`, `df.mode()`
- **Kiedy używać której miary** - wpływ wartości odstających (*outliers*), rozkłady skośne

Miary rozproszenia

- **Wariancja** - `np.var()`, `df.var()`
- **Odchylenie standardowe** - `np.std()`, `df.std()`
- **Rozstęp (range)** - `np.ptp()`, `df.max() - df.min()`
- **Kwartyle i IQR** - `np.percentile()`, `df.quantile()`
- **Współczynnik zmienności** - interpretacja rozproszenia

Prawdopodobieństwo i rozkłady

- **Rozkład normalny** - `scipy.stats.norm`
- **Rozkład jednostajny** - `scipy.stats.uniform`

- **Centralne twierdzenie graniczne** - praktyczne znaczenie
- **Wykrywanie rozkładu** - histogramy, Q-Q plots
- **Standardyzacja (z-score)** - `scipy.stats.zscore()`

Korelacja i zależności

- **Korelacja Pearsona** - `df.corr()`, `scipy.stats.pearsonr()`
- **Korelacja Spearmana** - `df.corr(method='spearman')`
- **Macierz korelacji** - `sns.heatmap()` wizualizacja
- **Interpretacja współczynników** - siła i kierunek związku
- **Korelacja vs przyczynowość** - podstawowe rozróżnienie

Regresja liniowa (podstawy)

- **Regresja prosta** - `scipy.stats.linregress()`
- **Współczynnik determinacji (R^2)** - jakość dopasowania
- **Interpretacja wyników** - slope, intercept, p-value
- **Wizualizacja** - `sns.regplot()`, `plt.scatter()` z linią trendu
- **Założenia regresji** - liniowość, normalność reszt

Testowanie hipotez (podstawy dla analityków)

- **Hipoteza zerowa i alternatywna** - formułowanie
- **P-value** - interpretacja i znaczenie
- **Test t-Studenta** - `scipy.stats.ttest_1samp()`, `ttest_ind()`
- **Test chi-kwadrat** - `scipy.stats.chi2_contingency()`
- **ANOVA** - `scipy.stats.f_oneway()` (podstawy)
- **Praktyczne zastosowania** - A/B testing, porównywanie grup

31. Praktyczne zastosowania statystyki w analizie danych

Eksploracyjna analiza danych (EDA) ze statystyką

- **Profilowanie danych** - automatyczne raporty z `pandas-profiling`
- **Wykrywanie anomalii** - metoda IQR, z-score
- **Analiza rozkładów** - skewness, kurtosis
- **Segmentacja danych** - analiza grup klientów
- **Benchmarking** - porównywanie wyników z normami

A/B Testing dla analityków

- **Projektowanie eksperymentu** - wielkość próby, randomizacja
- **Analiza wyników** - conversion rate, confidence intervals

- **Testy istotności** - czy różnica jest znacząca?
- **Praktyczne przykłady** - email campaigns, website changes
- **Błędy w A/B testing** - peeking, multiple testing

Raportowanie wyników statystycznych

- **Confidence intervals** - prezentowanie niepewności
- **Wizualizacja rozkładów** - boxploty, violin plots
- **Tabele kontyngencji** - `pd.crosstab()`
- **Komunikacja z biznesem** - jak tłumaczyć p-values
- **Dashboard ze statystykami** - kluczowe metryki

Etap 9: Dodatkowe narzędzia i biblioteki

Dlaczego? Bo warto znać różne narzędzia, żeby wiedzieć że są takie, które dają gotowe rozwiązania popularnych problemów.

32. SciPy - rozszerzone funkcje statystyczne

- **Instalacja**: `pip install scipy`
- **scipy.stats** - kompletna biblioteka statystyczna
- **Testy normalności** - Shapiro-Wilk, Kolmogorov-Smirnov
- **Testy nieparametryczne** - Mann-Whitney U, Wilcoxon
- **Funkcje rozkładów** - PDF, CDF, inverse CDF
- **Instalacja**: `pip install plotly`
- **Podstawowe wykresy**: `plotly.express`
- **Interaktywność**: zoom, hover, selection
- **Eksport**: HTML, PNG, PDF

34. Jupyter i produktywność

- **Magic commands**: `%time`, `%timeit`, `%prun`
- **Skróty klawiszowe** w Jupyter
- **Markdown** w notebook-ach
- **Eksport notebook-ów**: HTML, PDF, slides

35. Podstawy web scrapingu

- **requests** - pobieranie stron web
- **BeautifulSoup** - parsowanie HTML
- **Etyka scrapingu** - robots.txt, rate limiting

36. Praca z API

- **requests** - HTTP requests
- **JSON APIs** - pobieranie danych
- **Autentykacja** - API keys, OAuth
- **Przykłady**: Twitter API, Google APIs
- Uwaga - dużo o API piszę w **swojej książce**, o bibliotece requests też!

Praktyczne wskazówki do nauki

Strategia nauki

- **Praktyka codziennie** - chociaż 30-60 minut
- **Projekty praktyczne** - ucz się na rzeczywistych danych
- **Rozwiązuj problemy** - używaj gotowych platform: Kaggle Learn, HackerRank, LeetCode
- **Czytaj kod innych** - GitHub, Kaggle kernels
- **Dokumentacja** - naucz się czytać docs

Przydatne zasoby danych

- **Kaggle Datasets** - tysiące zestawów danych
- **UCI ML Repository** - klasyczne zestawy
- **Government Open Data** - dane publiczne
- **APIs**: Reddit, Twitter, GitHub, finansowe (np. NBP - o tym dokładnie jest moja książka!), pogodowe

Debugging i rozwiązywanie problemów

- **Czytaj komunikaty błędów** - są bardzo pomocne
- **print()** - najprościej debugowanie
- **Stack Overflow** - 99% problemów już rozwiązano
- **ChatGPT/Claude** - pomoc w wyjaśnianiu błędów

Kolejne kroki (po opanowaniu podstaw)

- **Machine Learning** - scikit-learn, podstawy ML
- **Deep Learning** - TensorFlow, PyTorch
- **Zaawansowana wizualizacja** - Streamlit, Plotly, D3.js, Dash
- **Big Data** - Polars, Spark, Dask
- **Deployment** - Flask, FastAPI, Docker
- **Cloud** - AWS, Google Cloud, Azure

Python - ćwiczenia

Etap 1: Środowisko pracy i podstawy

Ćwiczenie 1: Zainstaluj Python i Jupyter Notebook/JupyterLab

Uruchom swoje pierwsze **Jupyter Notebook** i wyświetl "Witaj, świecie!" za pomocą funkcji `print()`.

Ćwiczenie 2: Używaj Pythona jako kalkulatora

Wykonaj kilka podstawowych operacji matematycznych (dodawanie, odejmowanie, mnożenie, dzielenie, potęgowanie, reszta z dzielenia).

Ćwiczenie 3: Poćwicz komentarze i uruchamianie skryptów

Dodaj komentarze (#, """ """) do swojego kodu, aby wyjaśnić, co robią poszczególne linijki. Następnie stwórz plik `.py` z prostym kodem i uruchom go z terminala.

Ćwiczenie 4: Zapoznaj się z Condą/pip i środowiskami wirtualnymi

Stwórz swoje pierwsze środowisko wirtualne i zainstaluj w nim prostą bibliotekę (np. `numpy`).

Etap 2: Podstawy języka Python

Ćwiczenie 1: Zdefiniuj zmienne o różnych typach danych

Przypisz zmiennym liczby całkowite (`int`), zmiennoprzecinkowe (`float`), tekst (`str`) i wartości logiczne (`bool`). Wykonaj podstawowe operacje na tych typach (np. łączenie stringów, operacje arytmetyczne).

Ćwiczenie 2: Stwórz listę ulubionych filmów

Dodaj, usuń i zmodyfikuj elementy na liście. Poćwicz indeksowanie i slicing (`lista`, `lista[1:3]`). Spróbuj też **List comprehensions** do szybkiego tworzenia list.

Ćwiczenie 3: Zbuduj słownik z danymi klienta

Przechowuj imię, nazwisko, adres e-mail i numer telefonu jako pary klucz-wartość. Poćwicz dostęp do wartości i metody słowników (`keys()`, `values()`, `items()`).

Ćwiczenie 4: Napisz instrukcję warunkową

Sprawdź, czy wiek użytkownika kwalifikuje go do zakupu (np. `if wiek >= 18: ... else: ...`). Zastosuj `and`, `or`, `not`.

Ćwiczenie 5: Użyj pętli do przetwarzania danych

Wyświetl liczby od 1 do 10 za pomocą pętli `for` i `while`. Następnie, przejdź przez listę filmów i wyświetl każdy z nich. Poćwicz `break` i `continue`.

Ćwiczenie 6: Zdefiniuj funkcję

Zdefiniuj funkcję, która oblicza pole prostokąta, przyjmując długość i szerokość jako argumenty i zwracając pole. Dodaj argumenty domyślne.

Etap 3: Praca z danymi - biblioteki podstawowe

Ćwiczenie 1: Wczytaj i zapisz dane do pliku tekstowego

Otwórz plik (`open()`), zapisz do niego kilka linijek tekstu (`write()`), a następnie wczytaj je z powrotem (`read()`, `readlines()`). Użyj instrukcji `with`.

Ćwiczenie 2: Obsłuż błąd dzielenia przez zero

Napisz kod, który próbuje dzielić przez zero i użyj bloku **`try-except`** do przechwycenia błędu i wyświetlenia zrozumiałego komunikatu.

Ćwiczenie 3: Importuj moduły

Importuj moduł `datetime` i wyświetl aktualną datę i czas. Importuj moduł `os` i wyświetl aktualny katalog roboczy.

Ćwiczenie 4: Wczytaj dane z pliku CSV za pomocą modułu `csv`

Przejrzyj wiersze i kolumny danych. Zastanów się, jakie problemy z kodowaniem znaków mogą wystąpić (np. dla polskich znaków).

Etap 4: NumPy - obliczenia numeryczne

Ćwiczenie 1: Stwórz arrays z różnymi danymi

Utwórz array z liczbami całkowitymi, array z zerami i array z jedynkami. Porównaj wydajność operacji na dużej liście Pythona i dużym array NumPy.

Ćwiczenie 2: Wykonaj podstawowe operacje matematyczne na arrays

Dodaj dwa arrays, pomnóż array przez liczbę. Zobacz, jak działa **wektoryzacja** i **broadcasting**.

Ćwiczenie 3: Filtruj dane w array za pomocą boolean indexing

Stwórz array z ocenami studentów i wybierz tylko te oceny, które są powyżej 3.5.

Ćwiczenie 4: Oblicz statystyki dla danych w array

Wykorzystaj funkcje statystyczne NumPy takie jak `mean()`, `median()`, `std()`, `min()`, `max()`. Zastosuj operacje na osiach (`axis`).

Etap 5: Pandas - manipulacja danymi

Ćwiczenie 1: Wczytaj dane z pliku CSV do DataFrame

Znajdź publiczny zestaw danych (np. na Kaggle Datasets) i użyj `pd.read_csv()`. Poćwicz wczytywanie danych z Excela i JSON.

Ćwiczenie 2: Eksploruj wczytane dane

Użyj `head()`, `tail()`, `info()`, `describe()`, `shape`, `dtypes`, aby zrozumieć strukturę danych. Sprawdź unikalne wartości w kolumnach (`unique()`, `value_counts()`).

Ćwiczenie 3: Znajdź i uzupełnij brakujące wartości

Wykryj brakujące wartości za pomocą `isnull()` / `isna()` i spróbuj je uzupełnić (`fillna()`) lub usunąć (`dropna()`).

Ćwiczenie 4: Filtruj i wybieraj kolumny/wiersze

Wybierz tylko wybrane kolumny (`df[['kol1', 'kol2']]`), a następnie filtruj wiersze, aby spełniały określone warunki (np. sprzedaż > 1000) używając `loc[]` i `iloc[]`. Spróbuj też `query()`.

Ćwiczenie 5: Czyszczenie i transformacja danych

Usuń duplikaty (`drop_duplicates()`), zmień nazwy kolumn, zamień wartości (`replace()`). Zmień typy danych (`astype()`, `pd.to_datetime()`).

Ćwiczenie 6: Grupuj dane i agreguj wyniki

Zgrupuj dane sprzedażowe według produktu i oblicz sumę sprzedaży dla każdego produktu (`groupby()`, `sum()`). Poćwicz `agg()` z różnymi funkcjami agregującymi.

Ćwiczenie 7: Połącz dwa DataFrame'y

Stwórz dwa małe DataFrame'y, np. jeden z informacjami o klientach, drugi z ich zamówieniami, i połącz je za pomocą `merge()` (jak SQL JOIN).

Etap 6: Matplotlib - podstawowa wizualizacja danych

Ćwiczenie 1: Narysuj prosty wykres liniowy

Wyświetl dane o zmianach cen produktu w czasie.

Ćwiczenie 2: Stwórz histogram rozkładu wieku klientów

Zobacz, jak wyglądają rozkłady Twoich danych.

Ćwiczenie 3: Wizualizuj dane z DataFrame

Użyj `df.plot()` do szybkiego stworzenia wykresu słupkowego dla sumy sprzedaży według kategorii. Wypróbuj różne typy wykresów (`kind='line'`, `kind='bar'`).

Ćwiczenie 4: Dostosuj wykres

Dodaj tytuł, etykiety osi, legendę do swojego wykresu. Zmień kolory, style linii, rozmiar wykresu (`figsize`). Stwórz `subplots` (wiele wykresów na jednej figurze).

Etap 7: Seaborn - zaawansowana wizualizacja danych

Ćwiczenie 1: Stwórz bardziej estetyczny wykres słupkowy lub liniowy

Użyj Seaborn do wizualizacji danych o sprzedaży produktów. Zastosuj różne **style** Seaborn.

Ćwiczenie 2: Wizualizuj rozkład danych za pomocą `kdeplot` lub `histplot`

Zobacz gęstość rozkładu dla kolumny numerycznej.

Ćwiczenie 3: Stwórz macierz korelacji i wyświetl ją jako heatmapę

Zidentyfikuj silne korelacje między zmiennymi numerycznymi w Twoim DataFrame.

Ćwiczenie 4: Użyj `pairplot` do szybkiej eksploracji relacji między wieloma zmiennymi

Zobacz rozkłady pojedynczych zmiennych i korelacje między nimi na jednej siatce.

Ćwiczenie 5: Stwórz wykresy kategorialne

Użyj `boxplot` lub `violinplot` do porównania rozkładów wartości dla różnych kategorii.

Etap 8: Projekty praktyczne i zaawansowane techniki (EDA, Statystyka)

Ćwiczenie 1: Wykonaj eksploracyjną analizę danych (EDA) dla dowolnego zestawu danych z Kaggle

Zidentyfikuj brakujące wartości, typy danych, statystyki opisowe (`describe()`), a następnie stwórz wizualizacje rozkładów i korelacji. Wyciągnij z tego wnioski i "insights".

Ćwiczenie 2: Zastosuj miary tendencji centralnej i rozproszenia w Pythonie

Oblicz średnią, medianę, odchylenie standardowe i kwartyle dla kolumny numerycznej w swoim DataFrame (`df.mean()`, `df.median()`, `df.std()`, `df.quantile()`). Zinterpretuj wyniki w kontekście danych, zwracając uwagę na wpływ outlierów.

Ćwiczenie 3: Zbadaj korelację między zmiennymi

Oblicz macierz korelacji Pearsona i Spearmana (`df.corr()`). Stwórz heatmapę korelacji (`sns.heatmap()`). Zastanów się, czy korelacja oznacza **przyczynowość**.

Ćwiczenie 4: Wykonaj prosty test t-Studenta w Pythonie

Porównaj średnie dwóch grup (np. średnia sprzedaż przed i po kampanii marketingowej) i sprawdź istotność statystyczną za pomocą `scipy.stats.ttest_ind()`.

Ćwiczenie 5: Zinterpretuj wyniki prostego A/B testu

Na podstawie fikcyjnych danych, oblicz wskaźniki konwersji dla grupy A i B. Określ, czy różnica jest statystycznie istotna, bazując na p-value (zasymuluj lub znajdź przykład testu).

Etap 9: Dodatkowe narzędzia i biblioteki

Ćwiczenie 1: Użyj biblioteki `scipy.stats` do wykonania testu normalności

Sprawdź, czy Twoja kolumna danych ma rozkład normalny (np. test Shapiro-Wilka).

Ćwiczenie 2: Stwórz interaktywny wykres w Plotly

Wyświetl dane o czasie (szeregi czasowe) za pomocą Plotly i zobacz, jak działa interaktywność (zoom, hover, selection). Spróbuj wyeksportować wykres do pliku HTML.

Ćwiczenie 3: Wyślij proste zapytanie do API

Skorzystaj z biblioteki `requests`, aby pobrać dane z darmowego API (np. JSONPlaceholder lub OpenWeatherMap). Spróbuj pobrać dane JSON i przekształcić je na DataFrame Pandas.

Ćwiczenie 4: Wypróbuj podstawy web scrapingu

Spróbuj pobrać dane z prostej strony internetowej (np. listę filmów, publiczną tabelę) za pomocą `requests` i `BeautifulSoup`. **Pamiętaj o etyce scrapingu** (sprawdź `robots.txt` i limituj częstotliwość zapytań).

Python - quiz

Etap 1: Środowisko pracy i podstawy

Pytanie 1: Wymień dwa przykładowe narzędzia (IDE/edytory) idealne do analizy danych i eksploracji.

Odpowiedź 1: Idealne są Jupyter Notebook/JupyterLab lub Google Colab (środowisko w chmurze). Z popularnych programów instalowanych na komputerze: VS Code i PyCharm.

Pytanie 2: Jaką podstawową funkcję w Pythonie użyjesz do wyświetlenia informacji na ekranie?

Odpowiedź 2: Do wyświetlenia informacji służy funkcja print().

Etap 2: Podstawy języka Python

Pytanie 1: Wymień cztery podstawowe typy danych w Pythonie.

Odpowiedź 1: Podstawowe typy danych to int (liczby całkowite), float (liczby zmiennoprzecinkowe), str (tekst) oraz bool (wartości logiczne).

Pytanie 2: Czym różni się lista (list) od krotki (tuple) w Pythonie?

Odpowiedź 2: Lista jest zmienna (mutable), co oznacza, że można zmieniać jej elementy po utworzeniu, natomiast krotka jest niezmienna (immutable).

Pytanie 3: Do czego służy instrukcja if, elif, else w Pythonie?

Odpowiedź 3: Służą one do implementacji instrukcji warunkowych, pozwalając wykonać różne bloki kodu w zależności od spełnienia określonych warunków.

Pytanie 4: Jak zdefiniujesz prostą funkcję w Pythonie?

Odpowiedź 4: Funkcję definiuje się za pomocą słowa kluczowego def, np. def moja_funkcja(): pass.

Etap 3: Praca z danymi - biblioteki podstawowe (Obsługa plików, Błędy)

Pytanie 1: Jaką wbudowaną funkcją otworzysz plik w Pythonie? Wymień dwa podstawowe tryby otwierania plików.

Odpowiedź 1: Plik otworzysz funkcją open(). Podstawowe tryby to ‘r’ (odczyt) i ‘w’ (zapis).

Pytanie 2: Do czego służą bloki try, except w Pythonie?

Odpowiedź 2: Bloki try i except służą do obsługi błędów, pozwalając przechwycić i zareagować na wyjątki, które mogą wystąpić podczas wykonywania kodu.

Etap 4: NumPy - obliczenia numeryczne

Pytanie 1: Jakie jest główne zastosowanie biblioteki NumPy?

Odpowiedź 1: Głównym zastosowaniem NumPy jest efektywne wykonywanie obliczeń numerycznych, zwłaszcza na dużych tablicach (arrays).

Pytanie 2: Czym jest boolean indexing w NumPy?

Odpowiedź 2: Boolean indexing to technika filtrowania danych w tablicy NumPy przy użyciu tablicy wartości logicznych (True/False) o tym samym kształcie.

Pytanie 3: Wymień dwie przykładowe funkcje statystyczne dostępne w NumPy.

Odpowiedź 3: NumPy udostępnia funkcje takie jak mean() (średnia), median() (mediana), std() (odchylenie standardowe), min() czy max().

Etap 5: Pandas - manipulacja danymi

Pytanie 1: Jakie są dwie główne struktury danych w bibliotece Pandas?

Odpowiedź 1: Główne struktury to Series (jednowymiarowe dane z indeksem) oraz DataFrame (dwuwymiarowa tabela danych, przypominająca arkusz kalkulacyjny).

Pytanie 2: Jak wczytasz dane z pliku CSV do DataFrame w Pandas?

Odpowiedź 2: Dane z pliku CSV wczytasz za pomocą funkcji pd.read_csv().

Pytanie 3: Do czego służą metody isnull() lub isna() w Pandas?

Odpowiedź 3: Metody te służą do sprawdzania wartości brakujących (NULL/NaN) w danych, zwracając DataFrame o wartościach logicznych.

Pytanie 4: Jak zgrupujesz dane w DataFrame i obliczysz sumę dla każdej grupy?

Odpowiedź 4: Dane zgrupujesz za pomocą metody groupby(), a następnie zastosujesz funkcję agregującą, np. sum(), np. df.groupby('kolumna').sum().

Etap 6: Matplotlib - podstawowa wizualizacja danych

Pytanie 1: Jakie jest główne zastosowanie biblioteki Matplotlib?

Odpowiedź 1: Matplotlib służy do tworzenia statycznych wykresów i wizualizacji danych w Pythonie.

Pytanie 2: Jaki typ wykresu w Matplotlib najlepiej użyć do porównywania wartości między kategoriami?

Odpowiedź 2: Do porównywania wartości między kategoriami najlepiej użyć wykresu słupkowego (bar plot).

Etap 7: Seaborn - zaawansowana wizualizacja danych

Pytanie 1: Jakie jest główne zastosowanie biblioteki Seaborn w porównaniu do Matplotlib?

Odpowiedź 1: Seaborn jest biblioteką opartą na Matplotlib, która ułatwia tworzenie bardziej zaawansowanych i estetycznych wykresów statystycznych.

Pytanie 2: Jaki typ wykresu w Seaborn najlepiej nadaje się do wizualizacji korelacji między wieloma zmiennymi numerycznymi?

Odpowiedź 2: Do wizualizacji macierzy korelacji najlepiej użyć heatmapy z Seaborn (np. sns.heatmap()).

Etap 8: Projekty praktyczne i zaawansowane techniki (EDA, Statystyka)

Pytanie 1: Wymień trzy podstawowe kroki, które wykonasz podczas eksploracyjnej analizy danych (EDA).

Odpowiedź 1: EDA obejmuje m.in. wczytanie i pierwsze spojrzenie na dane, sprawdzenie jakości danych (np. brakujące wartości), obliczenie statystyk opisowych czy stworzenie wizualizacji rozkładów i korelacji.

Pytanie 2: Jakie jest podstawowe rozróżnienie między korelacją a przyczynowością?

Odpowiedź 2: Korelacja opisuje siłę i kierunek związku między zmiennymi, ale nie implikuje, że jedna zmienna powoduje drugą (przyczynowość).

Pytanie 3: Do czego służy test t-Studenta w statystyce i analizie danych?

Odpowiedź 3: Test t-Studenta służy do porównywania średnich między dwiema grupami lub porównania średniej próbki ze średnią populacji, aby ocenić, czy różnica jest statystycznie istotna. Jest używany np. w A/B testing.

Pytanie 4: Co oznacza p-value w kontekście testowania hipotez?

Odpowiedź 4: P-value to prawdopodobieństwo zaobserwowania wyników co najmniej tak ekstremalnych, jak uzyskane w badaniu, zakładając, że hipoteza zerowa (np. brak różnicy) jest prawdziwa. Niskie p-value (poniżej przyjętego progu, np. 0.05) sugeruje odrzucenie hipotezy zerowej.

Etap 9: Dodatkowe narzędzia i biblioteki (SciPy, Plotly, API)

Pytanie 1: Jakie jest główne zastosowanie biblioteki SciPy.stats?

Odpowiedź 1: SciPy.stats to kompletna biblioteka statystyczna w Pythonie, udostępniająca m.in. testy statystyczne (np. testy normalności, testy nieparametryczne) i funkcje rozkładów prawdopodobieństwa.

Pytanie 2: Czym charakteryzuje się biblioteka Plotly do wizualizacji danych?

Odpowiedź 2: Plotly pozwala na tworzenie interaktywnych wykresów, które można publikować w internecie (np. w formacie HTML).

Pytanie 3: Jaką biblioteką w Pythonie można wysyłać żądania do API w celu pobrania danych?

Odpowiedź 3: Do wysyłania żądań HTTP i pracy z API często używa się biblioteki requests.

Architektura rozwiązań - zupełne podstawy

Projektując rozwiązania (jaka data engineer) albo korzystając z nich (jako analityk danych/BI, nieco mniej jako data scientist) powinniśmy rozumieć skąd biorą się dane, jakie elementy składowe ekosystemów biorą udział w tych wszystkich przepływach. Zalecam zatem poznać zagadnienia, które tłumaczą:

Skąd biorą się dane?

- logi
- systemy transakcyjne
- e-commerce
- API

Gdzie dane są składowane?

- pliki
- bazy danych - różne rodzaje: SQL, NoSQL, key-value (Redis)
- hurtownie danych
- data lake

Jak przechodzą z jednego miejsca do drugiego?

- kopiowanie plików
- odczyt/zapis do tabel w bazach
- API
- streaming

Jakie są środowiska?

- chmura i on-prem - w kontekście infrastruktury
- dev, test, prod - w kontekście gdzie piszemy kod, gdzie testujemy, a co działa produkcyjnie

Jak to wszystko jest zorganizowane?

- kto zarządza procesami danych - orkiestracja, Apache Airflow
- procesy batchowe i real-time

Znajomość tych zagadnień pomaga analitykom i data scientistom lepiej rozumieć kontekst swojej pracy i efektywniej komunikować się z inżynierami danych.

Architektura rozwiązań - ćwiczenia

Etap 1: Skąd biorą się dane i gdzie są składowane?

Ćwiczenie 1: Zidentyfikuj źródła danych

Wymyśl scenariusz dla fikcyjnej firmy (np. platforma streamingowa) i wypisz, skąd mogą pochodzić dane (logi, systemy transakcyjne, e-commerce, API). Podaj konkretne przykłady dla każdego typu źródła.

Ćwiczenie 2: Zaproponuj miejsca składowania danych

Dla różnych typów danych z ćwiczenia 1 (np. dane transakcyjne, logi z aplikacji, dane z czujników IoT) zastanów się, gdzie najlepiej byłoby je przechowywać (relacyjne bazy SQL, bazy NoSQL takie jak Redis, hurtownie danych, data lake). Uzasadnij swój wybór.

Ćwiczenie 3: Rysuj proste schematy architektury

Narysuj prosty schemat przepływu danych od źródła (np. aplikacja mobilna) do miejsca składowania (np. baza danych), a następnie do systemu analitycznego (np. hurtownia danych).

Etap 2: Jak przechodzą z miejsca na miejsce i jak są zorganizowane?

Ćwiczenie 1: Opisz proces ETL/ELT

Wyobraź sobie, że musisz przenieść dane o zamówieniach z systemu transakcyjnego do hurtowni danych. Opisz, jakie kroki (Extract, Transform, Load) byś podjął, i jakie narzędzia (np. Python + SQL, Power Query) mogłyby w tym pomóc.

Ćwiczenie 2: Porównaj przetwarzanie batchowe i real-time

Zastanów się, kiedy użyłbyś przetwarzania danych w paczkach (batch), a kiedy w czasie rzeczywistym (streaming). Podaj przykłady zastosowań dla obu (np. dane o sprzedaży vs dane z czujników temperatury, czy wykrywanie oszustw).

Ćwiczenie 3: Zrozum środowiska pracy

Wyjaśnij, czym różnią się środowiska Dev, Test i Prod w kontekście rozwoju projektu danych. Zastanów się, dlaczego są one potrzebne i jakie różnice w konfiguracji mogłyby między nimi występować.

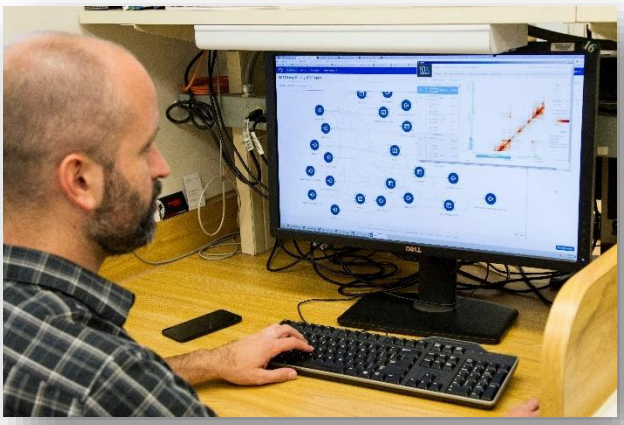
Ćwiczenie 4: Rola orkiestracji

Zastanów się, dlaczego narzędzia takie jak **Apache Airflow** są potrzebne w złożonych przepływach danych. Wymyśl prosty przepływ zadań (np. pobierz plik, przetwórz, załaduj do bazy, wyślij raport) i zastanów się, jak byś go zaplanował.

Ścieżki specjalizacji w dziedzinie danych

Po opanowaniu podstaw Python + SQL + Excel + statystyki, można wybrać jedną z głównych ścieżek specjalizacji:

Data Scientist



Koncentruje się na budowaniu modeli predykcyjnych, odkrywaniu wzorców, eksperymentach i algorytmach machine learning. Wymaga silnych podstaw matematycznych i statystycznych oraz zaawansowanego programowania. Jest najtrudniejszą ścieżką wejścia na rynek ze względu na wysokie wymagania. Przejście na tę ścieżkę wymaga rozwinięcia umiejętności w Machine Learning, Deep Learning, zaawansowanej statystyce i modelowaniu matematycznym, przetwarzaniu języka naturalnego (NLP) i analizie obrazów. **Przykładowe projekty** obejmują predykcję cen nieruchomości (dokładnie o tym jest mój darmowy kurs - zapisz się), klasyfikację spamu, systemy rekomendacji filmów, analizę sentymentu recenzji i predykcję churnu (odejść) klientów.

Wybierz Data Science jeśli

- ✅ Lubisz matematykę i statystykę
- ✅ Interesują Cię algorytmy i modelowanie
- ✅ Chcesz budować systemy predykcyjne
- ✅ Jesteś cierpliwy (długie eksperymenty)
- ✅ Lubisz research i odkrywanie wzorców

Typowe projekty

- **Predykcja cen nieruchomości** - regresja z feature engineering
- **Klasyfikacja spam-u email** - NLP + classification
- **Rekomendacje filmów** - collaborative filtering

- **Analiza sentymentu recenzji** - NLP end-to-end
- **Predykcja churn klientów** - business-focused classification

Data Engineer

Zajmuje się budowaniem i utrzymywaniem infrastruktury danych, tworzeniem pipeline'ów ETL/ELT i zarządzaniem hurtowniami danych. Wymaga doświadczenia z systemami i jest ścieżką o średniej trudności wejścia na rynek. Ale poza wiedzą o danych przydatna, o ile nie wymagana jest wiedza z zakresu DevOps - wdrażanie rozwiązań, budowanie przepływów nie tylko danych, ale też wdrażających kod. **Projekty** często dotyczą automatyzacji pipeline'ów danych i integracji wielu źródeł.

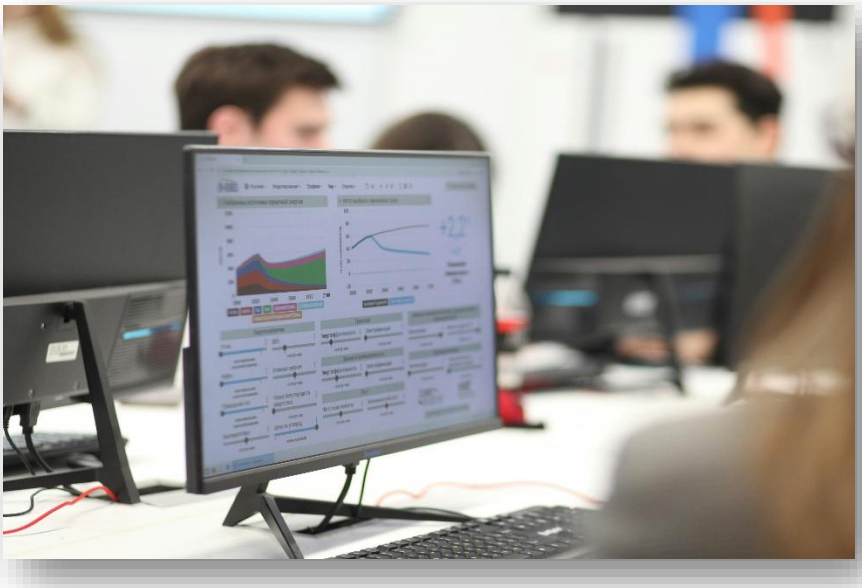
Wybierz Data Engineering jeśli

- ✅ Lubisz systemowe myślenie
- ✅ Interesują Cię infrastruktura i architektura
- ✅ Chcesz pracować z big data
- ✅ Lubisz automatyzację i optymalizację
- ✅ Nie przeszkadza Ci praca "za kulisami"

Typowe projekty

- **End-to-end ETL pipeline** - z API do data warehouse
- **Real-time streaming system** - Kafka + Spark Streaming
- **Data lake architecture** - S3 + Glue + Athena
- **Dockerized data pipeline** - z CI/CD
- **Multi-source data integration** - różne formaty i źródła

Business Intelligence (BI) Analyst



Skupia się na raportowaniu, tworzeniu dashboardów i dostarczaniu *business insights*. Wymaga intensywnej współpracy z biznesem i jest najłatwiejszą ścieżką wejścia na rynek. Projekty obejmują przygotowanie dashboard pokazujących m.in. sprzedaż, analizy kampanii marketingowych, zestawienia finansowe itd.

Wybierz BI Analysis jeśli

- ✅ Lubisz kontakt z biznesem
- ✅ Chcesz szybko widzieć rezultaty pracy
- ✅ Interesuje Cię storytelling z danymi
- ✅ Lubisz wizualizacje i dashboardy
- ✅ Chcesz bezpośrednio wpływać na decyzje biznesowe

Typowe projekty

- **Executive dashboard** - high-level KPIs z drill-down
- **Sales performance tracker** - pipeline, territories, reps
- **Financial reporting suite** - P&L, balance sheet, cash flow
- **Marketing campaign analyzer** - multi-channel attribution
- **Operational metrics dashboard** - real-time monitoring

W kolejnych rozdziałach omawiam dokładnie ścieżki rozwoju dla każdej z ról oraz typowe projekty.

Data Scientist - profil

Główne zadania i obszar działania

Specjalista ds. uczenia maszynowego koncentruje się na **tworzeniu modeli predykcyjnych, odkrywaniu ukrytych wzorców w danych oraz prowadzeniu eksperymentów analitycznych**, które wspierają decyzje biznesowe. To rola łącząca kompetencje techniczne, statystyczne i analityczne – ale także wymagająca umiejętności interpretacji wyników i przekładania ich na język zrozumiały dla decydentów.

Typowe produkty pracy data scientista

- **Modele uczenia maszynowego** – np. do prognozowania popytu, klasyfikacji klientów, wykrywania anomalii czy personalizacji ofert
- **Algorytmy i prototypy** – niestandardowe rozwiązania dostosowane do specyfiki danych i problemów w organizacji
- **Wnioski i rekomendacje biznesowe (insights)** – ustrukturyzowane odpowiedzi na pytania „dlaczego?" i „co jeśli?", często w formie wizualizacji, raportów lub prezentacji
- **Eksperymenty A/B** – testy hipotez w oparciu o dane rzeczywiste, wspierające rozwój produktów cyfrowych (np. aplikacji, serwisów)

Współpraca i interakcje z innymi rolami

Data scientist współpracuje z różnymi osobami w zależności od etapu projektu i rodzaju problemu:

- **Z zespołami produktowymi** – aby zrozumieć potrzeby użytkowników i dostarczać dane wspierające rozwój funkcji lub rekomendacji
- **Z biznesem i interesariuszami** – w celu tłumaczenia wyników analiz, przedstawiania przewidywań oraz wspólnego wyciągania wniosków
- **Czasem z inżynierami danych lub DevOps** – aby wdrożyć modele do produkcji lub zapewnić dostęp do danych potrzebnych do trenowania

Podsumowanie

Data scientist to specjalista zajmujący się analizą danych na wyższym poziomie – nie tylko **opisuje przeszłość**, ale przede wszystkim **prognozuje przyszłość** i pomaga podejmować decyzje w oparciu o dane. Jego praca łączy **algorytmy, statystykę i zrozumienie kontekstu biznesowego**. To rola wymagająca nie tylko wiedzy technicznej, ale także umiejętności przekładania złożonych modeli na konkretne działania i rekomendacje.

Data Scientist - plan rozwoju

Etap 1: Machine Learning

Teoria i podstawy

- **Uczenie nadzorowane** (supervised learning): klasyfikacja vs regresja
- **Uczenie nienadzorowane** (unsupervised learning): grupowanie (clustering), redukcja wymiarowości
- **Nadmierne dopasowanie/niedostateczne dopasowanie**: kompromis między odchyleniem a wariancją
- **Cross-validation**: podział próbki na train/validation/test
- **Szukanie dodatkowych cech** (feature engineering): selekcja, transformacja cech

Scikit-learn - praktyka

- **Klasyfikacja**: Logistic Regression, Random Forest, SVM
- **Regresja**: Linear Regression, Decision Trees, Ensemble methods
- **Clustering**: K-means, DBSCAN, Hierarchical
- **Preprocessing danych**: StandardScaler, LabelEncoder, Pipeline
- **Ocena modelu**: dokładność, precyzja, odwołanie, F1, AUC-ROC

Etap 2: Zaawansowana statystyka

- **Statystyka bayesowska**: a priori, a posteriori, prawdopodobieństwo
- **Statystyka wielowymiarowa**: ANOVA, MANOVA
- **Analiza szeregów czasowych**: ARIMA, rozkład sezonowy
- **Projekt eksperymentalny**: randomizacja, analiza mocy
- **Analiza przeżycia**: regresja Kaplana-Meiera, Coxa

Etap 3: Deep Learning

TensorFlow/Keras lub PyTorch

- **Sieci neuronowe**: perceptron, propagacja wsteczna
- **Regularyzacje**: porzucanie, normalizacja wsadowa
- **CNN**: przetwarzanie obrazów, widzenie komputerowe
- **RNN/LSTM**: dane sekwencyjne, szeregi czasowe
- **Uczenie transferowe**: modele wstępnie wytrenowane

Etap 4: Specjalizacje

Natural Language Processing - przetwarzanie języka naturalnego

- **NLTK/spaCy**: tokenizacja, tagowanie POS, NER
- **Wstępne przetwarzanie tekstu**: czyszczenie, stemming, lematyzacja
- **Ekstrakcja cech**: TF-IDF, osadzanie słów (Word2Vec, GloVe)
- **Analiza sentymentów**: klasyfikacja tekstu
- **Modelowanie tematów**: LDA, NMF
- **Transformatory**: BERT, GPT (podstawy)

Computer Vision - widzenie komputerowe, praca z obrazem

- **OpenCV**: podstawy przetwarzania obrazu
- **Wstępne przetwarzanie obrazu**: powiększanie, normalizacja
- **Architektury CNN**: LeNet, AlexNet, ResNet
- **Wykrywanie obiektów**: YOLO, R-CNN (podstawy)
- **Klasyfikacja obrazu**: praktyczne projekty

Time Series & Forecasting - szeregi czasowe i ich przewidywanie

- **Statsmodels**: ARIMA, SARIMA, rozkład sezonowy
- **Prophet**: narzędzie prognozujące od Facebooka/Meta
- **Zaawansowane metody**: sieci typu LSTM dla szeregów czasowych
- **Zastosowania biznesowe**: prognozowanie sprzedaży, planowanie popytu

Etap 5: MLOps i produkcja

- **Wersjonowanie modeli**: MLflow, DVC
- **Wdrażanie modeli**: Flask API, podstawy Dockera
- **Monitorowanie modeli**: wykrywanie dryfu, śledzenie wydajności
- **Testowanie A/B** dla modeli ML
- **Platformy chmurowe**: AWS SageMaker, Google AI Platform (podstawy)

Data Scientist - typowe projekty

Projekty pokrywają różne obszary biznesowe

- **Nieruchomości** (Predykcja cen) - wycena i inwestycje
- **Cyberbezpieczeństwo** (Klasyfikacja spamu) - automatyzacja filtracji
- **Media/Entertainment** (Rekomendacje) - personalizacja i engagement
- **E-commerce** (Analiza sentymentu) - customer feedback
- **Retention** (Predykcja churn) - zarządzanie klientami

Pokazują różne umiejętności techniczne

- **Regresja i feature engineering** (ceny nieruchomości)
- **NLP i przetwarzanie tekstu** (spam, sentyment)
- **Systemy rekomendacji** i collaborative filtering
- **Deep learning** (LSTM, transformers)
- **MLOps i produkcja** (monitoring, deployment)

Demonstrują znajomość biznesu

- **KPI nieruchomości** - cena/m^2, ROI, analiza lokalizacji
- **Metryki bezpieczeństwa** - współczynnik fałszywych wyników pozytywnych, dokładność wykrywania
- **Metryki zaangażowania** - współczynnik retencji, dokładność rekomendacji
- **Wgląd w klienta** - wyniki sentymentu, postrzeganie marki
- **Wpływ na biznes** - współczynnik odejść, CLV, koszty retencji

1. Predykcja cen nieruchomości – regresja z feature engineering

Co to jest

Model regresyjny przewidujący cenę nieruchomości na podstawie jej cech (lokalizacja, metraż, liczba pokoi itd.). Projekt łączy klasyczne techniki modelowania z intensywnym feature engineeringiem. **Zobacz gotowy, zrealizowany projekt**

Kluczowe komponenty

- **Eksploracja danych**: rozkłady cen, korelacje zmiennych
- **Czyszczenie danych**: brakujące dane, outliery, standaryzacja
- **Feature engineering**:
  - Ceny za m^2
  - Odległość do centrum / przystanków
  - Jakość dzielnicy (np. scoring z danych publicznych)
- **Modelowanie**:
  - Modele bazowe: regresja liniowa, drzewiaste (XGBoost, LightGBM)
  - Walidacja krzyżowa, tuning hiperparametrów
- **Ewaluacja**: RMSE, MAE, mapa błędów

Przykładowe źródła danych

- Kaggle - Boston House Prices
- Otwarte dane miejskie (np. Warszawa, Kraków – ceny m^2, lokalizacje)
- OLX, Otodom – dane webscrapowane

Wyzwania techniczne

- Zmienność cen w zależności od mikro-lokalizacji
- Kategoryzacja zmiennych tekstowych (np. stan: „do remontu” vs „idealny”)
- Feature leakage – unikanie wprowadzania przyszłości do modelu

Technologie

- Python (pandas, scikit-learn, XGBoost)
- Jupyter Notebook
- GeoPandas, Folium (dla map lokalizacji)

2. Klasyfikacja spamu – NLP + klasyfikacja

Co to jest

System klasyfikujący wiadomości e-mail jako spam lub nie-spam na podstawie ich treści.

Kluczowe komponenty

- **Przygotowanie danych tekstowych**:
  - Czyszczenie: usuwanie HTML, znaków specjalnych
  - Tokenizacja i stemming/lemmatyzacja
- **Feature extraction**:
  - TF-IDF
  - Word embeddings (opcjonalnie: FastText/BERT)
- **Modelowanie**:
  - Modele klasyfikacyjne: Naive Bayes, SVM, Random Forest
- **Ewaluacja**:
  - Accuracy, Precision, Recall, F1
  - Macierz pomyłek

Przykładowe źródła danych

- SMS Spam Collection Dataset (UCI)
- Enron Email Dataset

Wyzwania techniczne

- Niezbalansowane klasy – spam to tylko część danych
- Tokenizacja wielojęzyczna / emoji / nietypowe znaki
- Overfitting przy dużej liczbie słów

Technologie

- Python (NLTK, spaCy, scikit-learn)
- Jupyter Notebook
- sklearn-pipeline do automatyzacji preprocessing + model

3. Rekomendacje filmów – collaborative filtering

Co to jest

System rekomendacji oparty na zachowaniach użytkowników – proponuje filmy na podstawie podobnych użytkowników (collaborative filtering).

Kluczowe komponenty

- **Macierz użytkownik-film**: oceny, kliknięcia, obejrzenia
- **Podejścia rekomendacyjne**:
  - User-based i item-based collaborative filtering
  - Alternatywnie: matrix factorization (SVD, ALS)
- **Cold start problem** – nowi użytkownicy/filmy
- **Ewaluacja**: Precision@k (ile trafnych wyników w top-k), Recall@k (ile trafnych udało się znaleźć), MAP (średnia precyzji dla wszystkich trafień), NDCG (trafność z uwzględnieniem pozycji)

Przykładowe źródła danych

- MovieLens
- IMDb (z scrapingiem lub API)
- Netflix Prize Dataset

Wyzwania techniczne

- Bardzo rzadkie dane (sparsity matrix)
- Rekomendacje muszą być szybkie (cache?)
- Skalowalność – tysiące użytkowników i filmów

Technologie

- Python (surprise, lightFM, pandas)
- Jupyter Notebook
- Streamlit/Gradio do prototypu interfejsu

4. Analiza sentymentu recenzji – NLP end-to-end

Co to jest

System do klasyfikacji sentymentu recenzji (np. pozytywna/negatywna). Przykład realnego zastosowania NLP w biznesie (np. ecommerce).

Kluczowe komponenty

- **Pipeline NLP**:
  - Preprocessing: czyszczenie, tokenizacja, lematyzacja
  - Embedding: TF-IDF, GloVe, Word2Vec, BERT
- **Modelowanie**:
  - Prosty: Logistic Regression / SVM
  - Zaawansowany: LSTM / Transformer (HuggingFace)
- **Interpretacja wyników**:
  - LIME / SHAP – co wpływa na predykcję
  - Word clouds

Przykładowe źródła danych

- IMDb Reviews
- Amazon product reviews (dostępne na Kaggle)
- Recenzje z Allegro, OLX – do scrapowania

Wyzwania techniczne

- Ironia, sarkazm – trudne dla modeli klasycznych
- Obiektywne vs subiektywne zdania
- Długość i jakość tekstu – bardzo różna

Technologie

- Python (spaCy, HuggingFace Transformers, sklearn)
- TensorFlow/Keras lub PyTorch
- Jupyter Notebook

5. Predykcja churn klientów – klasyfikacja z kontekstem biznesowym

Co to jest

Model przewidujący, którzy klienci mogą odejść (churn) w najbliższym czasie, bazując na ich aktywności, cechach i historii zakupów.

Kluczowe komponenty

- **Analiza eksploracyjna**:
  - Identyfikacja cech odróżniających klientów churnujących
- **Feature engineering**:
  - RFM (Recency, Frequency, Monetary)
  - Aktywność: logowania, zakupy, kontakt z supportem
- **Modelowanie**:
  - Drzewa decyzyjne, XGBoost, Random Forest
  - Analiza cech – co wpływa na churn
- **Ewaluacja**:
  - F1, ROC-AUC, precision/recall
  - Wartość biznesowa – np. ile utraconego przychodu można przewidzieć

Przykładowe źródła danych

- Telco Churn Dataset
- Dane firmowe (CRM, e-commerce, SaaS logi)

Wyzwania techniczne

- Niezbalansowane dane – churn to często <10%
- Zmienność cech w czasie – np. wzorzec aktywności się zmienia
- Powiązanie z działaniami marketingowymi

Technologie

- Python (pandas, scikit-learn, XGBoost)
- Tableau / Power BI do pokazania wyników
- MLflow do trackowania modeli

Data Scientist - aspekty techniczne

Narzędzia do tworzenia

- **Python** – biblioteki takie jak pandas, scikit-learn, TensorFlow lub PyTorch
- **Jupyter Notebook** – eksploracja danych i tworzenie prototypów
- **Git** – system kontroli wersji i współpraca zespołowa
- **Docker** – tworzenie kontenerów z modelami do wdrożeń
- **MLflow** – śledzenie eksperymentów i wersjonowanie modeli

Architektura uczenia maszynowego

- **Źródła danych** – interfejsy API, bazy danych, pliki CSV lub JSON
- **Przetwarzanie danych** – czyszczenie, przygotowanie, inżynieria cech
- **Uczenie modelu** – walidacja krzyżowa, dostrajanie parametrów
- **Ocena modelu** – metryki, walidacja, testowanie
- **Wdrożenie** (Deployment) – punkty dostępu (API), monitorowanie, aktualizacje

Dobre praktyki

- **Powtarzalność** – ustawianie losowości, zarządzanie środowiskiem
- **Jakość kodu** – czytelność, dokumentacja, testy
- **Walidacja modelu** – podział na zbiory uczący, walidacyjny i testowy, walidacja krzyżowa (cross-validation)
- **Inżynieria cech** (Feature Engineering) – wiedza dziedzinowa, automatyczny wybór cech
- **Gotowość do produkcji** – obsługa błędów, logowanie, monitorowanie

Umiejętności potrzebne do stworzenia

- **Python** – znajomość zaawansowanych bibliotek do uczenia maszynowego i głębokiego
- **Statystyka** – zrozumienie algorytmów i ich założeń
- **Wiedza dziedzinowa** – rozumienie kontekstu biznesowego
- **Inżynieria oprogramowania** – czysty kod, testowanie, wdrożenia
- **Komunikacja** – prezentacja wyników, opowiadanie historii z użyciem danych

Data Scientist - ćwiczenia

Mini-projekt 1: Ulepszanie modelu predykcji cen nieruchomości

- **Cel:** Zbuduj lub ulepsz model przewidujący ceny nieruchomości, bazując na ich cechach.
- **Kroki:** Pobierz dane (np. Kaggle: Boston House Prices, lub dane z OLX/Otodom – web scraping). Wykonaj **Feature Engineering**, tworząc nowe, potencjalnie lepsze zmienne (np. cena za m^2, odległość do centrum handlowego/szkoły). Zbuduj **model regresyjny** (np. Regresja Liniowa, Drzewa Decyzyjne, XGBoost z `scikit-learn`) i oceń jego jakość (RMSE, MAE). Użyj Jupyter Notebook.
- **Pobudzanie wyobraźni:** Jakie inne dane zewnętrzne (np. przestępczość w okolicy, dostępność komunikacji miejskiej, dane demograficzne) mogłyby poprawić Twój model? Jakie lokalne portale mogłyby być źródłem danych (web scraping)?

Mini-projekt 2: Prosty klasyfikator spamu

- **Cel:** Stwórz system, który automatycznie rozróżnia wiadomości spam od wiadomości niebędących spamem.
- **Kroki:** Pobierz zestaw danych (np. SMS Spam Collection Dataset z UCI). Wykonaj **wstępne przetwarzanie tekstu** (czyszczenie, tokenizacja, stemming/lematyzacja). Zastosuj **ekstrakcję cech** (np. TF-IDF). Zbuduj **model klasyfikacyjny** (np. Naive Bayes, SVM, Random Forest z `scikit-learn`) i oceń go za pomocą metryk takich jak Accuracy, Precision, Recall, F1 oraz macierzy pomyłek.
- **Pobudzanie wyobraźni:** Jakie wyzwania pojawią się przy dłuższych tekstach (np. e-mailach firmowych)? Jak radzić sobie z emotikonami, slangiem czy językiem potocznym w kontekście wykrywania spamu?

Mini-projekt 3: System rekomendacji dla fikcyjnego sklepu

- **Cel:** Zaproponuj, jakie produkty kupiłby klient, bazując na jego wcześniejszych zakupach lub zakupach podobnych klientów (collaborative filtering).
- **Kroki:** Użyj publicznego zestawu danych (np. MovieLens, ale zinterpretuj go jako zakupy produktów). Zastosuj podejście **collaborative filtering** (user-based lub item-based). Zastanów się nad **cold start problemem** (co polecić nowemu użytkownikowi lub nowy produkt). Oceń swój system rekomendacji za pomocą metryk takich jak Precision@k, Recall@k.
- **Pobudzanie wyobraźni:** Jak system rekomendacji wpływa na sprzedaż? Jakie inne dane (np. oceny, przeglądane produkty, czas spędzony na stronie) mogłyby wzbogacić rekomendacje?

Mini-projekt 4: Analiza sentymentu recenzji produktowych

- **Cel:** Klasyfikuj recenzje produktów (np. z fikcyjnego sklepu e-commerce) jako pozytywne, negatywne lub neutralne, aby zrozumieć opinie klientów.
- **Kroki:** Pobierz recenzje (np. Amazon product reviews dostępne na Kaggle). Zbuduj **pipeline NLP** (preprocessing tekstu: czyszczenie, tokenizacja, lematyzacja; embedding: TF-IDF, Word2Vec, lub podstawy BERT z HuggingFace Transformers). Wytrenuj **model klasyfikacyjny** (np. Logistic Regression, SVM, lub prostą sieć neuronową). Zinterpretuj wyniki za pomocą narzędzi takich jak LIME/SHAP.
- **Pobudzanie wyobraźni:** Jakie korzyści biznesowe płyną z automatycznej analizy sentymentu dla firmy e-commerce (np. szybka reakcja na negatywne opinie, identyfikacja problemów z produktem)? Jak rozpoznać ironię lub sarkazm w tekście?

Data Scientist - quiz

Etap 1: Machine Learning

Pytanie 1: Jaka jest główna różnica między supervised learning a unsupervised learning?

Odpowiedź 1: W supervised learning modele uczą się na danych zawierających zarówno cechy, jak i odpowiadające im etykiety (prawidłowe odpowiedzi) (np. klasyfikacja, regresja). W unsupervised learning modele pracują na danych bez etykiet, szukając wzorców lub struktur (np. clustering).

Pytanie 2: Co to jest problem overfitting i jak pomaga w jego uniknięciu cross-validation?

Odpowiedź 2: Overfitting to sytuacja, gdy model zbyt dobrze dopasowuje się do danych treningowych, tracąc zdolność generalizacji na nowe dane. Cross-validation pomaga ocenić wydajność modelu na niewidzianych danych poprzez podział zbioru danych na wiele podzbiorów (np. foldów) i wielokrotne trenowanie i testowanie modelu, co daje bardziej rzetelny obraz błędu generalizacji.

Pytanie 3: Wymień dwie przykładowe metryki ewaluacji dla modelu klasyfikacyjnego.

Odpowiedź 3: Metryki ewaluacji dla klasyfikacji to np. accuracy (dokładność) , precision (precyzja), recall (czułość), F1 czy AUC-ROC.

Etap 2: Zaawansowana statystyka

Pytanie 1: Wymień jeden przykładowy obszar zaawansowanej statystyki kluczowy dla data scientisty, wykraczający poza podstawy.

Odpowiedź 1: Przykładami są statystyka bayesowska, statystyka wielowymiarowa (np. ANOVA), analiza szeregów czasowych (np. ARIMA) czy analiza przeżycia.

Etap 3: Deep Learning

Pytanie 1: Jakie są dwa główne typy sieci neuronowych wymienione w planie rozwoju data scientisty, stosowane odpowiednio do przetwarzania obrazów i sekwencji danych?

Odpowiedź 1: Do przetwarzania obrazów i computer vision stosuje się CNN (Convolutional Neural Networks). Do pracy z sekwencjami danych, takimi jak szeregi czasowe, stosuje się RNN (Recurrent Neural Networks) lub LSTM.

Pytanie 2: Do czego służy technika transfer learning w Deep Learningu?

Odpowiedź 2: Transfer learning polega na wykorzystaniu wcześniej wytrenowanego modelu (np. na dużym zbiorze danych) jako punktu wyjścia do rozwiązania nowego, podobnego problemu, co pozwala zaoszczędzić czas i dane treningowe.

Etap 4: Specjalizacje (np. Natural Language Processing)

Pytanie 1: W ramach Przetwarzania Języka Naturalnego (NLP), czym zajmuje się analiza sentymentu?

Odpowiedź 1: Analiza sentymentu to proces klasyfikacji tekstu (np. recenzji, postów) w celu określenia wyrażanej w nim opinii lub emocji (np. pozytywna, negatywna, neutralna).

Pytanie 2: Wymień dwie techniki ekstrakcji cech z tekstu stosowane w NLP.

Odpowiedź 2: Technikami ekstrakcji cech są np. TF-IDF (Term Frequency-Inverse Document Frequency) lub word embeddings (np. Word2Vec, GloVe, a także te z nowszych modeli jak BERT).

Data Engineer - profil

Główne zadania i obszar działania

Inżynier danych koncentruje się na **budowie, rozwijaniu i utrzymywaniu infrastruktury danych**, która umożliwia zbieranie, przetwarzanie, przechowywanie i udostępnianie danych w organizacji. Jego praca stanowi fundament dla zespołów analitycznych i zajmujących się uczeniem maszynowym, ponieważ to właśnie dzięki dobrze zaprojektowanym rozwiązaniom inżynierii danych możliwe jest skuteczne wykorzystanie danych do analiz, prognoz czy automatyzacji procesów. Czyli: to właśnie inżynier danych dostarcza danych do dalszych analiz i dba o to, aby były na czas i dobrej jakości.

Typowe produkty pracy inżyniera danych

- **Procesy ETL / ELT** *(Extract, Transform, Load / Extract, Load, Transform)* – automatyczne pipeline'y do pobierania danych z różnych źródeł, ich przekształcania i ładowania do systemów docelowych
- **Hurtownie danych (data warehouses)** – scentralizowane systemy przechowywania danych w ustrukturyzowanej formie, zoptymalizowane pod kątem zapytań analitycznych
- **Systemy przetwarzania strumieniowego (real-time)** – rozwiązania do analizy danych "na żywo", wykorzystywane np. do monitorowania zachowań użytkowników, wykrywania oszustw czy alertowania w czasie rzeczywistym
- **Zarządzanie jakością i spójnością danych** – tworzenie reguł walidacyjnych, testów integralności, wykrywanie braków lub duplikatów danych

Współpraca i interakcje z innymi rolami

Inżynier danych pracuje na styku technologii i analizy, dlatego na co dzień współpracuje z wieloma osobami i zespołami:

- **Z analitykami danych i data scientistami** – aby dostarczać im uporządkowane, wysokiej jakości dane do analiz i budowy modeli predykcyjnych
- **Z zespołami DevOps lub inżynierii oprogramowania** – w celu wdrażania i monitorowania pipeline'ów danych w środowiskach produkcyjnych
- **Z przedstawicielami biznesu lub właścicielami produktów** – zwłaszcza przy definiowaniu źródeł danych, reguł przekształceń lub wymagań raportowych

Podsumowanie

Inżynier danych to specjalista od fundamentów ekosystemu danych – odpowiada za to, by dane były **dostępne, kompletne, spójne i aktualne**. Jego praca jest niewidoczna na pierwszy rzut oka, ale bez niej nie byłoby możliwe ani tworzenie raportów, ani trenowanie modeli uczenia maszynowego, ani prowadzenie analiz wspierających decyzje biznesowe.

Data Engineer - plan rozwoju

Etap 1: Zaawansowany SQL i bazy danych

Zaawansowane SQL

- **Window functions** – funkcje okienkowe, operacje na partycjach danych
- **Rekurencyjne CTE** – zapytania do danych hierarchicznych (np. struktura organizacyjna)
- **Optymalizacja zapytań** – analiza planów wykonania, indeksowanie, eliminacja wąskich gardeł
- **Procedury składowane** – automatyzacja logiki przetwarzania w bazie danych
- **Transakcje** – zasady ACID, poziomy izolacji i spójności danych

Typy baz danych

- **OLTP vs OLAP** – różnice między bazami transakcyjnymi a analitycznymi
- **Podstawy NoSQL** – wprowadzenie do MongoDB, Redis (struktur danych, przypadki użycia)
- **Hurtownie danych** – modelowanie wymiarowe, schemat gwiazdy
- **Bazy danych w chmurze** – podstawy BigQuery, Redshift, Snowflake

Etap 2: Python w inżynierii danych

Zaawansowany Python

- **Programowanie asynchroniczne** – obsługa wielu zadań równolegle (pakiety takie jak asyncio, aiohttp)
- **Wieloprocesowość** – przetwarzanie równoległe z użyciem `concurrent.futures`, `joblib`
- **Obsługa błędów** – tworzenie odpornych na błędy pipeline'ów
- **Logowanie** – strukturalne logowanie zdarzeń i danych
- **Testowanie** – automatyczne testy komponentów z użyciem `pytest`

Biblioteki inżynierskie

- **SQLAlchemy** – biblioteka ORM do pracy z bazami danych
- **Apache Airflow** – orkiestracja zadań, budowanie DAG-ów
- **Dask** – przetwarzanie danych w dużej skali w Pythonie (równolegle i rozproszone)
- **Great Expectations** – automatyczne testy jakości danych

Etap 3: Platformy chmurowe

Na początek najlepiej wybierz jedną platformę (AWS / GCP / Azure) i się na niej skup:

Amazon Web Services (AWS)

- **S3** – przechowywanie obiektowe (*object storage*), fundament data lake
- **RDS / Aurora** – zarządzane bazy relacyjne
- **Redshift** – hurtownia danych w chmurze
- **Glue** – usługa ETL w chmurze
- **Lambda** – bezserwerowe funkcje (serverless)
- **Kinesis** – strumieniowe przetwarzanie danych w czasie rzeczywistym

Google Cloud Platform (GCP)

- **BigQuery** – bezserwerowa hurtownia danych
- **Cloud Storage** – przechowywanie plików i danych binarnych
- **Dataflow** – potok danych oparty na Apache Beam
- **Cloud Functions** – funkcje bezserwerowe
- **Pub/Sub** – kolejki wiadomości i przetwarzanie zdarzeń
- **Cloud SQL** – zarządzane bazy relacyjne

Microsoft Azure

- **Azure Data Factory** – tworzenie potoków ETL/ELT
- **Azure Synapse** – platforma do analiz i pracy z dużymi zbiorami danych
- **Blob Storage** – magazyn danych obiektowych
- **Azure SQL** – zarządzana baza danych SQL
- **Event Hubs** – przetwarzanie strumieni danych

Etap 4: Technologie Big Data

Apache Spark

- **PySpark** – interfejs Pythona do Spark
- **Operacje na DataFrame** – transformacje i akcje
- **Spark SQL** – zapytania SQL w środowisku Spark
- **Przetwarzanie strumieniowe** – podstawy Spark Streaming
- **Optymalizacja wydajności** – partycjonowanie, buforowanie danych (caching)

Ekosystem Hadoop (podstawy)

- **HDFS** – rozproszony system plików
- **Hive** – wykonywanie zapytań SQL nad danymi w Hadoop

- **Formaty kolumnowe** – Parquet, ORC
- **Delta Lake** – warstwa zarządzająca spójnością i wersjonowaniem danych w jeziorze danych

Dodatkowo

Przyda się znajomość technologii:

- **Apache Kafka** - strumieniowe przesyłanie wiadomości (zdarzeń), zagadnienie producenta i konsumenta wiadomości
- **Kafka Connect**, **Kafka Streams** - jako rozszerzenie wiedzy o Kafce
- **Apache Flink** - technologia przetwarzania strumieniowego i batchowego, kod pisany w Javie i Scali, są też biblioteki dla Pythona
- **Scala** - jako język przygotowany do zadań big data

Etap 5: DevOps dla danych

Infrastruktura jako kod

- **Docker** – konteneryzacja aplikacji i procesów
- **Docker Compose** – definiowanie i uruchamianie aplikacji wielokontenerowych
- **Terraform** – zarządzanie infrastrukturą chmurową jako kodem (podstawy)
- **Kubernetes** – orkiestracja kontenerów, podstawowe pojęcia i wdrożenia

CI/CD (ciągła integracja i dostarczanie)

- **Git workflows** – strategie rozgałęzień dla kodu (np. Git Flow, trunk-based)
- **GitHub Actions / GitLab CI** – automatyzacja testów i wdrożeń
- **Jakość kodu** – lintowanie, formatowanie, testy jednostkowe
- **Strategie wdrożeń** – blue-green, canary deployments

Etap 6: Monitorowanie i obserwowalność

- **Monitorowanie aplikacji** – Prometheus, Grafana
- **Agregacja logów** – zestaw ELK (Elasticsearch, Logstash, Kibana)
- **Śledzenie przepływu danych (data lineage)** – identyfikacja źródeł i zależności
- **Systemy alertów** – integracje z PagerDuty, Slackiem
- **Monitorowanie kosztów** – optymalizacja wydatków w chmurze

Data Engineer - typowe projekty

Podsumowanie projektów

- **End-to-End ETL Pipeline** - Pokazuje podstawowe umiejętności integracji danych
- **Real-time Streaming** - Demonstruje pracę z big data i systemami czasu rzeczywistego

- **Data Lake Architecture** - Architektura danych w chmurze (AWS/GCP/Azure)
- **Dockerized Pipeline** - DevOps, CI/CD, monitoring, production-ready solutions
- **Multi-source Integration** - Najbardziej złożony, enterprise-level projekt

Kluczowe kompetencje

Techniczne

- **Python** wraz z bibliotekami pandas, SQLAlchemy, asyncio
- **SQL** na poziomie zaawansowanym
- **Platformy chmurowe** - AWS/GCP/Azure
- **Konteneryzacja** - Docker, Kubernetes
- **Przetwarzanie strumieniowe** (*Streaming*) - Kafka, Spark Streaming
- **Monitorowanie systemów** - Prometheus, Grafana

Architektura systemów

- **Modelowanie danych** - schemat gwiazdy, jeziora danych
- **Optymalizacja wydajności** - dzielenie na fragmenty, przetwarzanie równoległe
- **Jakość danych** - walidacja, monitorowanie, alertowanie
- **Skalowalność** - obsługa milionów rekordów

Biznesowe

- **Integracja międzydziałowa** - łączenie systemów różnych departamentów
- **Zarządzanie danymi** (*Data governance*)- bezpieczeństwo, zgodność z przepisami, śledzenie pochodzenia danych (*lineage*)
- **Optymalizacja kosztów** - efektywne wykorzystanie zasobów chmurowych

1. End-to-End ETL Pipeline – Kompletny rurociąg danych od API do hurtowni

Co to jest

Zautomatyzowany pipeline, który pobiera dane z różnych źródeł (głównie API), przekształca je zgodnie z regułami biznesowymi, a następnie ładuje do hurtowni danych w odpowiednim schemacie. System wspiera analitykę, BI i dalsze przetwarzanie danych.

Architektura ETL

Extract – Pobieranie danych

Źródła danych:

- **REST API** – Salesforce (CRM), Google Ads, X/Twitter, Facebook
- **Bazy danych** – PostgreSQL, MySQL z systemów transakcyjnych
- **Pliki** – CSV/JSON z SFTP, chmury (S3, Google Drive)
- **Web scraping** – dane publiczne (z poszanowaniem robots.txt) **Przykładowe API:**
- OpenWeatherMap, Alpha Vantage, GitHub API, News API, JSONPlaceholder (testy)

Transform – Przekształcanie danych

Czyszczenie i standaryzacja:

- Usuwanie duplikatów, uzupełnianie braków (null handling)
- Walidacja formatów (e-mail, daty, numery)
- Wykrywanie odstających wartości (outliers)
- Konwersje typów danych (np. string → date) **Logika biznesowa:**
- Obliczenia wskaźników (ROI, marże, wzrost)
- Agregacje czasowe (dzień/miesiąc/rok)
- Łączenie danych po kluczach (joins)
- Normalizacja (waluty, kraje, daty)
- Wzbogacenie (geolokalizacja, segmentacja klientów)

Load – Ładowanie do hurtowni danych

Modelowanie danych:

- **Fakty (fact tables)** – transakcje, zdarzenia
- **Wymiary (dimension tables)** – produkty, klienci, czas
- **SCD (Slowly Changing Dimensions)** – śledzenie zmian danych w czasie
- **Data marts** – gotowe widoki dla zespołów marketingu, sprzedaży itd.

Warstwa techniczna

Kluczowe komponenty

- **Orkiestracja** – Apache Airflow (DAG-i harmonogramujące zadania)
- **Monitoring i alerty** – logi, metryki, alerty SLA, retry logic
- **Quality assurance** – testy danych w Great Expectations
- **Bezpieczeństwo** – klucze API w HashiCorp Vault / AWS Secrets

Przykład zastosowania – E-commerce Analytics

- **Extract**: Shopify API (zamówienia), Google Ads API (kampanie), Facebook API (reklamy)
- **Transform**: Łączenie kampanii z zamówieniami, wyliczenie wskaźników ROI, CLTV
- **Load**: PostgreSQL z modelem gwiazdy
- **Consume**: Dashboard Power BI dla zespołu marketingu

2. Real-time Streaming System – System strumieniowy w czasie rzeczywistym

Co to jest

System do przetwarzania i analizy danych „na żywo", w momencie ich pojawiania się. Zamiast przetwarzać dane w batchach (np. co godzinę lub dzień), system umożliwia natychmiastową reakcję na zdarzenia z różnych źródeł.

Architektura Apache Kafka + Spark Streaming

Warstwa komunikacji (Apache Kafka)

Źródła danych (producenci):

- **Website events** – kliknięcia, odsłony, zakupy z Google Analytics lub własnego trackingu
- **IoT sensors** – dane z czujników: temperatura, wilgotność, GPS
- **Application logs** – błędy i metryki wydajności aplikacji
- **Change Data Capture (CDC)** – zmiany w PostgreSQL/MySQL (np. Debezium)
- **Social media** – dane z API X/Twittera, Instagrama (np. hashtagi, wzmianki)

Warstwa przetwarzania (Apache Spark Streaming)

Real-time analytics:

- **Windowed aggregations** – np. liczba eventów w ostatnich 5 minutach
- **Wykrywanie wzorców** – identyfikacja anomalii, sekwencji zachowań
- **Ocena ryzyka i scoring** – np. wykrywanie fraudów w czasie rzeczywistym
- **Enrichment** – wzbogacanie danych w locie (np. dodanie danych referencyjnych)
- **Complex event processing** – analiza ciągów zdarzeń użytkowników

Warstwa docelowa (Output sinks)

- **Dashboardy real-time** – Grafana, Kibana
- **Systemy powiadomień** – Slack, PagerDuty
- **Modele ML online** – scoring, aktualizacje modeli
- **Hurtownie danych** – BigQuery, PostgreSQL (wyniki agregacji)
- **Wyszukiwarki** – Elasticsearch do natychmiastowego wyszukiwania

Przypadki użycia

E-commerce – Personalizacja w czasie rzeczywistym

- **Kliknięcie produktu** wyzwala event w Kafka
- **Spark aktualizuje profil użytkownika** i oblicza podobne produkty
- **Rekomendacje aktualizowane** są bezpośrednio na stronie
- **Monitoring A/B testów** - natychmiastowy wgląd w konwersje

Finanse – Wykrywanie fraudów

- **Transakcja płatnicza** → Kafka event
- **Spark analizuje dane historyczne** → ocena ryzyka
- **Wynik scoringu** → w czasie <100ms
- **Decyzja** → akceptacja / odrzucenie / weryfikacja ręczna

IoT – Smart Building

- **Dane z czujników** → temperatura, ruch, zużycie energii
- **Spark agreguje dane** → np. średnia temperatura na piętrze
- **Reakcja systemu** → regulacja klimatyzacji, alert bezpieczeństwa, na przykład w wyniku reakcji na event z Kafki
- **Dashboard facility management** → aktualizowany w czasie rzeczywistym

Techniczne wyzwania

- **Wysoka przepustowość** – miliony zdarzeń na sekundę
- **Niskie opóźnienie** – <1 sekundy od zdarzenia do reakcji
- **Odporność na awarie** – przetwarzanie mimo problemów z węzłami
- **Exactly-once semantics** – żadne zdarzenie nie może być przetworzone wielokrotnie
- **Obsługa przeciążenia (backpressure)** – regulacja tempa przetwarzania przy dużym napływie danych

3. Data Lake Architecture – Architektura jeziora danych

Co to jest

Centralne repozytorium danych dla całej organizacji, umożliwiające przechowywanie danych w surowej (*raw*) formie oraz ich późniejsze przetwarzanie w celach analitycznych, raportowych i uczenia maszynowego. Dane trafiają do systemu w różnych formatach, a warstwa katalogująca i zapytań umożliwia wygodne ich eksplorowanie.

Architektura jeziora danych na AWS (S3 + Glue + Athena)

Warstwa przechowywania (Amazon S3)

Obsługiwane formaty plików:

- **Parquet** – kolumnowy format zoptymalizowany pod analitykę
- **Delta Lake** – obsługa wersjonowania i transakcji ACID
- **JSON** – dane półstrukturalne z API
- **Avro** – ewolucja schematów, popularne przy streamingu
- **CSV** – prosty format, często z systemów legacy i do debugowania

Warstwa katalogowania i zarządzania schematami (AWS Glue)

- **Automatyczne wykrywanie danych** – AWS Glue skanuje pliki w S3 i generuje schematy
- **Katalog schematów** – centralne miejsce dla definicji tabel i pól
- **Śledzenie pochodzenia danych (data lineage)** – informacje o źródłach i transformacjach danych
- **Klasyfikacja danych** – wykrywanie danych wrażliwych (np. dane osobowe)
- **Partycjonowanie danych** – przyspieszanie zapytań poprzez podział np. po dacie (rok/miesiąc/dzień)

Silnik zapytań (Amazon Athena)

- **Ad-hoc analytics** – wykonywanie zapytań SQL bez konieczności ładowania danych do bazy
- **Integracja z Glue Catalog** – pełna zgodność ze strukturą katalogu
- **Obsługa dużych wolumenów danych** – efektywna analiza danych bez potrzeby ich przemieszczania

Zarządzanie danymi i bezpieczeństwo

Kontrola dostępu

- Uprawnienia do katalogów i plików (IAM, S3 bucket policies)
- Separacja środowisk (np. dev/test/prod) w obrębie jeziora danych

Jakość danych i monitoring

- **AWS Glue Data Quality** – testy jakości danych bezpośrednio na DataFrame'ach
- **Amazon CloudWatch** – monitorowanie wydajności, kosztów, błędów przetwarzania
- **AWS Macie** – automatyczne wykrywanie danych wrażliwych (np. PESEL, numery kart)
- **Optymalizacja kosztów** – klasy przechowywania S3 (Standard, Infrequent Access, Glacier), polityki retencji

Przypadki użycia

Media – Analityka treści

- **Dane o oglądalności i interakcjach** ładowane z aplikacji mobilnych i serwisów streamingowych
- **Transformacja i katalogowanie** w Glue
- **Eksploracja danych** w Athena – raporty treści o największym zaangażowaniu
- **Zasilenie dashboardów BI** i rekomendacji

Retail – Widok 360 klienta

- **Łączenie danych z e-commerce, CRM i kampanii marketingowych**
- **Wzbogacenie danych o dane demograficzne i geograficzne**
- **Analiza zachowań klientów w czasie rzeczywistym**
- **Zasilenie modeli predykcyjnych (churn, rekomendacje)**

4. Dockerized Data Pipeline – Skonteneryzowany rurociąg danych z CI/CD

Co to jest

System ETL oparty na Dockerze, zautomatyzowany przez CI/CD. Pipeline'y są uruchamiane w kontenerach, co zapewnia powtarzalność, niezawodność, łatwe testowanie i wdrażanie w różnych środowiskach (dev/test/prod).

Architektura kontenerowa (Docker + docker-compose)

Warstwa kontenerów (multi-container setup)

- **ETL application** – główny kontener z logiką ekstrakcji, transformacji i ładowania
- **Scheduler** – harmonogram zadań (np. cron/Airflow)
- **Database** – kontener z bazą danych pomocniczą (np. PostgreSQL)
- **Monitoring** – Prometheus + Grafana do obserwacji działania pipeline'ów

Dockerfile dla aplikacji ETL

- Konfiguracja środowiska
- Instalacja zależności (Python, pandas, SQLAlchemy itd.)
- Punkt wejścia do uruchamiania zadań

Automatyzacja CI/CD (GitHub Actions)

- **Testy jednostkowe i integracyjne** przy każdym pushu kodu
- **Testy jakości kodu** - Sonar
- **Budowanie obrazów Docker** i publikacja do rejestru (np. Docker Hub) po każdym merge kodu
- **Wdrażanie pipeline'ów** na serwery testowe i produkcyjne

Strategia testowania

Testy jakości danych

- **Great Expectations** – walidacja typów, zakresów, reguł spójności

Testy integracyjne

- Sprawdzenie działania całego pipeline'u od źródła do celu
- Testy z fałszywymi danymi produkcyjnymi

Monitoring i obserwowalność

Metryki pipeline'u

- Czas przetwarzania, rozmiar danych, liczba błędów
- Logowanie z poziomu aplikacji (np. do ELK stack)

Dashboardy

- **Grafana** – wizualizacja metryk i alerty (np. czas trwania pipeline > N minut)

5. Multi-source Data Integration – Integracja danych z wielu źródeł

Co to jest

System integrujący dane z różnorodnych źródeł – relacyjnych baz danych, plików, API i źródeł strumieniowych. Celem jest ujednolicenie schematów, konsolidacja danych i udostępnienie spójnego widoku dla zespołów analitycznych i produktów danych.

Architektura integracji danych

Warstwa zbierania danych (Data Ingestion Layer)

- **Batch ingestion** – pliki CSV, Excel, eksporty z baz
- **API ingestion** – dane z REST API (np. systemy CRM, ERP)
- **CDC ingestion** – zmiany w bazach (PostgreSQL, MySQL)
- **Streaming ingestion** – dane w czasie rzeczywistym (Kafka, dane z IoT np. przez MQTT)

Zarządzanie schematami (Schema Management)

- Automatyczne dopasowywanie pól i typów danych
- Walidacja spójności i wersjonowanie schematów

Transformacja i harmonizacja

- Mapowanie pól z różnych źródeł do jednego modelu logicznego
- Standaryzacja jednostek, formatów, języków
- Wzbogacanie danych referencyjnych (np. ISO kody krajów, słowniki branżowe)

Różnorodne źródła danych

Dane strukturalne

- **PostgreSQL, MySQL, SQL Server, Oracle** – źródła transakcyjne
- **Amazon RDS, Google BigQuery, Snowflake, Azure SQL** – hurtownie danych w chmurze

Dane półstrukturalne i plikowe

- **NoSQL** – MongoDB, Redis (pamięć podręczna, JSON-y)
- **Pliki** – CSV, Excel, pliki logów
- **Streaming** – Kafka (logi, zdarzenia aplikacyjne, dane telemetryczne), MQTT

Przypadki użycia

Hurtownia danych

- Konsolidacja danych z 10+ źródeł do jednego modelu analitycznego
- Ładowanie do BigQuery i Snowflake

System raportowy

- Zasilanie Power BI/Looker dashboardów
- Zapytania łączące dane historyczne z bieżącymi

Integracja systemów operacyjnych

- Połączenie systemu HR (SQL Server) z CRM (API REST) i aplikacją do rozliczeń (Oracle)
- Synchronizacja danych w czasie rzeczywistym i wsadowo

Data Engineer - aspekty techniczne

Narzędzia do tworzenia systemów danych

- **Apache Airflow** - program do zarządzania przepływem pracy i automatyzacji zadań
- **Apache Spark** - narzędzie do przetwarzania ogromnych ilości danych
- **Apache Kafka** - system do przesyłania danych w czasie rzeczywistym
- **Terraform** - narzędzie do automatycznego tworzenia infrastruktury w chmurze za pomocą kodu
- **Docker** - technologia do "pakowania" aplikacji w pojemniki (kontenery), żeby działały wszędzie tak samo
- **Kubernetes** - system do zarządzania wieloma pojemnikami Docker na raz, automatycznie je uruchamia, restartuje gdy się zepsują, rozdziela ruch między nimi i skaluje w górę lub w dół w zależności od potrzeb. Innymi słowy: jeśli Docker to "pojedynczy pojemnik z aplikacją", to Kubernetes to "inteligentny zarządca całego magazynu pełnego takich pojemników" - pilnuje żeby wszystko działało, zastępuje zepsute pojemniki nowymi i dodaje więcej gdy jest duży ruch.

Architektura (budowa) systemów danych

- **Źródła danych** - skąd pochodzą dane: bazy danych, pliki, strony internetowe, aplikacje poprzez API
- **Warstwa pobierania** - jak dane są zbierane i wczytywane do systemu: *batch* vs *streaming*
- **Warstwa przechowywania** - gdzie dane są składowane (np. w chmurze Amazon S3, hurtowni Google BigQuery)
- **Warstwa przetwarzania** - gdzie dane są czyszczone, przekształcane, agregowane i wzbogacane
- **Warstwa konsumpcji** - jak gotowe dane są udostępniane użytkownikom (raporty, dashboardy, aplikacje wystawiające API, wykorzystanie danych przez modele uczenia maszynowego)

Dobre praktyki

- **Skalowalność** - system musi radzić sobie z rosnącą ilością danych
- **Niezawodność** - system musi działać stabilnie, nawet gdy coś pójdzie nie tak
- **Monitorowanie** - stałe sprawdzanie czy wszystko działa poprawnie oraz odpowiedni alerty
- **Bezpieczeństwo** - ochrona danych i kontrola dostępu

- **Optymalizacja kosztów** - efektywne wykorzystanie zasobów, żeby nie płacić za więcej niż potrzeba (szczególnie w chmurze)

Umiejętności potrzebne do stworzenia takich systemów

- **Python/SQL** - zaawansowane programowanie do manipulacji danymi
- **Platformy chmurowe** - znajomość usług AWS/Google Cloud/Microsoft Azure na średnim poziomie
- **DevOps** - umiejętność wdrażania i zarządzania aplikacjami
- **Big Data** - praca z wielkimi zbiorami danych i systemami rozproszonymi - przyda się Spark, znajomość Kafki, może Scala albo Flink
- **Projektowanie systemów** - planowanie architektury, która będzie wydajna i skalowalna

Data Engineer - ćwiczenia

Mini-projekt 1: Prosty ETL Pipeline

- **Cel:** Zbuduj automatyczny przepływ danych, który pobiera dane z publicznego API lub pliku, przekształca je zgodnie z regułami biznesowymi i ładuje do lokalnej bazy danych.
- **Kroki:** Pobierz dane z publicznego API (np. OpenWeatherMap) lub pliku CSV. Użyj Pythona (biblioteka `requests` do API, `pandas` do transformacji) do ekstrakcji i transformacji. Następnie użyj `SQLAlchemy` do ładowania danych do lokalnej bazy PostgreSQL. Zadbaj o podstawowe **czyszczenie i standaryzację danych**. Zautomatyzuj to za pomocą `cron` (w systemie Linux) lub podstaw **Apache Airflow** (jeśli masz zainstalowany).
- **Pobudzanie wyobraźni:** Jak zapewnić, że dane są zawsze aktualne i spójne? Jakie testy można dodać, aby sprawdzać jakość danych w pipeline? Jakie narzędzia (np. `Great Expectations`) mogłyby pomóc w testowaniu jakości danych?

Mini-projekt 2: Monitorowanie danych w czasie rzeczywistym (symulacja)

- **Cel:** Zasymuluj system, który przetwarza "zdarzenia" na żywo i wyświetla zagregowane wyniki, np. w terminalu.
- **Kroki:** Użyj Pythona do generowania symulowanych "zdarzeń" (np. kliknięć na stronie internetowej, odczytów z czujników temperatury). Zapisuj je do pliku lub wysyłaj do prostej kolejki (np. Redis lub symulacji kolejki w Pythonie). Napisz skrypt Pythona, który "konsumuje" te zdarzenia, agreguje je w krótkich oknach czasowych (np. co 5 sekund) i wyświetla wynik. Zastanów się nad przypadkami użycia, takimi jak wykrywanie fraudów w finansach.
- **Pobudzanie wyobraźni:** Jakie alerty można by dodać, gdy liczba zdarzeń przekroczy pewien próg lub wartości anomalne zostaną wykryte? Jakie dane z IoT (Internetu Rzeczy) mogłyby być przetwarzane w ten sposób (np. poziom wody, zużycie energii)?

Mini-projekt 3: Ustrukturyzowane przechowywanie danych w Data Lake (symulacja)

- **Cel:** Zbuduj prostą strukturę Data Lake, przechowując dane w surowej formie, a następnie w przetworzonej, gotowej do analizy.
- **Kroki:** Stwórz na swoim dysku lokalnym foldery symulujące warstwy Data Lake: `raw` i `processed`. Umieść w folderze `raw` pobrane pliki CSV bez obróbki. Napisz skrypt Pythona (`pandas`), który wczytuje te pliki, czyści je, przekształca do bardziej analitycznego formatu (np. symulując format kolumnowy taki jak Parquet lub Delta

Lake, zapisując jako CSV z odpowiednio ustrukturyzowanymi kolumnami), i zapisuje do folderu `processed`.

- **Pobudzanie wyobraźni:** Jak to wyglądałoby w chmurze (np. AWS S3 jako magazyn obiektowy)? Jak zapewnić, że schemat danych jest spójny w warstwie `processed` (np. za pomocą narzędzi takich jak AWS Glue Data Catalog)?

Mini-projekt 4: Konteneryzacja prostej aplikacji danych

- **Cel:** Użyj **Docker**, aby spakować prosty skrypt Pythona, który przetwarza dane, w przenośny kontener.
- **Kroki:** Napisz prosty skrypt Pythona, który wczytuje plik (np. CSV), coś z nim robi (np. filtruje, agreguje), i zapisuje wynik do innego pliku. Stwórz **Dockerfile**, który zawiera wszystkie zależności i instrukcje do uruchomienia tego skryptu. Zbuduj obraz Docker i uruchom kontener.
- **Pobudzanie wyobraźni:** Jak używać **Docker Compose**, aby uruchomić aplikację Pythona wraz z bazą danych (np. PostgreSQL) w osobnych kontenerach? Jakie korzyści daje konteneryzacja dla niezawodności i powtarzalności środowiska? Jak można by to wdrażać automatycznie z pomocą **CI/CD** (np. GitHub Actions)?

Data Engineer - quiz

Etap 1: Rozszerzone SQL i bazy danych

Pytanie 1: Wymień jedną zaawansowaną technikę SQL, która pozwala na wykonywanie obliczeń na grupach wierszy powiązanych ze sobą w oknie.

Odpowiedź 1: Taką techniką są Funkcje okna (Window Functions).

Pytanie 2: Czym różnią się bazy danych typu OLTP od OLAP?

Odpowiedź 2: Bazy OLTP (Online Transaction Processing) są zoptymalizowane do szybkiego przetwarzania wielu małych transakcji (np. systemy transakcyjne), podczas gdy bazy OLAP (Online Analytical Processing) są zoptymalizowane do złożonych zapytań analitycznych na dużych zbiorach danych (np. hurtownie danych).

Etap 2: Python dla inżynierii danych

Pytanie 1: Wymień jedną bibliotekę Pythonową stosowaną do orkiestracji (zarządzania przepływem) zadań i potoków danych.

Odpowiedź 1: Przykładem jest biblioteka Apache Airflow lub Prefect.

Pytanie 2: Do czego służy biblioteka Great Expectations w kontekście inżynierii danych?

Odpowiedź 2: Great Expectations służy do testowania jakości danych i walidacji danych w potokach.

Etap 3: Systemy ETL/ELT

Pytanie 1: W klasycznym procesie ETL, co oznaczają litery E, T i L?

Odpowiedź 1: E oznacza Extract (wyciąganie danych ze źródeł), T oznacza Transform (transformacja i czyszczenie danych), a L oznacza Load (ładowanie danych do celu, np. hurtowni danych).

Pytanie 2: Wymień dwie przykładowe techniki czyszczenia danych w fazie Transformacji.

Odpowiedź 2: Technikami czyszczenia są np. usuwanie duplikatów (Deduplication), walidacja formatów danych, obsługa pustych wartości (Null handling) czy konwersja typów danych.

Etap 4: Technologie Big Data

Pytanie 1: Jakie jest główne zastosowanie platformy Apache Spark?

Odpowiedź 1: Apache Spark służy do przetwarzania danych na dużą skalę, zarówno w trybie batch, jak i strumieniowym.

Pytanie 2: Wymień jeden z kolumnowych formatów plików często używanych w ekosystemie Big Data, takich jak Hadoop lub Spark.

Odpowiedź 2: Przykładem jest format Parquet lub ORC, które są zoptymalizowane do zapytań analitycznych.

Etap 5: DevOps dla danych

Pytanie 1: Do czego służy narzędzie Docker?

Odpowiedź 1: Docker służy do konteneryzacji aplikacji, co pozwala na pakowanie kodu, bibliotek i zależności w przenośne kontenery, ułatwiając wdrażanie - na przykład potoków danych.

Pytanie 2: Jakie jest zastosowanie narzędzia Terraform?

Odpowiedź 2: Terraform służy do Infrastructure as Code (IaC), co pozwala na zarządzanie infrastrukturą (np. serwerami, bazami danych w chmurze) za pomocą kodu, a nie ręcznej konfiguracji.

BI Analyst (analityk biznesowy) - profil roli

Główne zadania i obszar działania

Analityk biznesowy koncentruje się na **raportowaniu, tworzeniu dashboardów oraz dostarczaniu wniosków biznesowych na podstawie danych**. To rola blisko związana z codziennym funkcjonowaniem organizacji – BI analyst odpowiada za to, by osoby decyzyjne miały zawsze aktualny i zrozumiały obraz sytuacji operacyjnej i strategicznej, aby mogły podejmować decyzje na podstawie danych (*data driven*).

Typowe produkty pracy analityka BI

- **Interaktywne dashboardy** – wizualizacje danych w narzędziach takich jak Power BI, Tableau czy Looker, które umożliwiają użytkownikom eksplorację danych na własną rękę
- **Raporty cykliczne** – regularne zestawienia (np. tygodniowe, miesięczne), śledzące kluczowe wskaźniki (KPI) i trendy biznesowe; najczęściej generowane batchowo, na przykład za pomocą Spark
- **Analizy ad hoc** – jednorazowe opracowania na potrzeby konkretnych pytań lub decyzji (np. analiza spadku sprzedaży w danym regionie); generowane jednorazowo, ale warto zachować kod - przyda się więc znajomość Git
- **Rekomendacje i podsumowania** – interpretacje danych dostarczane w przystępnej formie, często w prezentacjach lub raportach PDF; tu przydają się narzędzia łączące tekst opisowy z kodem i jego wynikiem (tabelami, wykresami) typu **Quarto**

Współpraca i interakcje z innymi rolami

Analityk BI działa **bardzo blisko biznesu** i interesariuszy – jego rola wymaga nie tylko znajomości danych, ale także zrozumienia celów, procesów i wyzwań danej organizacji:

- **Z przedstawicielami biznesu (sprzedaż, marketing, operacje)** – w celu zbierania wymagań, tłumaczenia danych na działania i wspierania decyzji
- **Z właścicielami produktów i menedżerami** – w zakresie monitorowania wskaźników i oceny efektywności działań
- **Czasem z inżynierami danych** – aby upewnić się, że dane są dostępne, spójne i odpowiednio przygotowane do analizy

Podsumowanie

Analityk biznesowy to **łącznik między danymi a decyzjami biznesowymi**. Jego zadaniem jest nie tylko raportowanie „co się dzieje?", ale również pomaganie w zrozumieniu „dlaczego?" i „co dalej?". To rola analityczna, ale mocno osadzona w rzeczywistości biznesowej – wymaga zarówno umiejętności technicznych, jak i komunikacyjnych.

BI Analyst - plan rozwoju

Etap 1: Zaawansowane narzędzia BI

Wybierz jedno narzędzie jako główne, drugie jako uzupełniające.

Power BI (ekosystem Microsoft)

- **Power BI Desktop** – tworzenie raportów
- **DAX (Data Analysis Expressions)** – kolumny obliczeniowe, miary
- **Power Query** – transformacja danych, język M
- **Modelowanie danych** – relacje, schemat gwiazdy
- **Power BI Service** – publikowanie i udostępnianie raportów on-line, współpraca
- **Gateway** – połączenia z danymi lokalnymi (on-premises)

Tableau

- **Tableau Desktop** – arkusze, dashboardy, historie
- **Kolumny obliczeniowe** – podstawowe i zaawansowane kalkulacje
- **Parametry i filtry** – interaktywność
- **Połączenia danych** – różnorodne źródła
- **Tableau Server/Online** – publikacja i współdzielenie
- **Optymalizacja wydajności** – ekstrakty, agregacje

Etap 2: Modelowanie danych dla BI

Modelowanie wymiarowe

- **Schemat gwiazdy vs schemat płatka śniegu**
- **Tabele faktów vs tabele wymiarów**
- **Powoli zmieniające się wymiary (Slowly Changing Dimensions, SCD)** - przechowywanie wersji historycznych danych
- **Data marts** – wyodrębnione podzbiory tematyczne
- **Metodyka Kimballa** – podejście bottom-up przy budowaniu hurtowni danych - zaczynasz od tego co naprawdę potrzebujesz i stopniowo rozbudowujesz:
  - najpierw konkretne potrzeby biznesowe, a nie ogólna teoria
  - hurtownia ma być łatwa w użyciu dla analityków i menedżerów
  - robimy małymi krokami, dodając kolejne obszary biznesowe

Wydajność i optymalizacja

- **Strategie indeksowania** – pod raportowanie

- **Partycjonowanie** – duże tabele faktów
- **Tabele agregacji** – pre-liczone podsumowania
- **Incremental refresh** – ładowanie tylko nowych lub zmienionych danych

Etap 3: Biznesowa domena wiedzy

Poznaj wymienione hasła, nazwy wskaźników - zorientuj się do czego są używane, z czym się je je. Nie musisz znać ich na pamięć - ważne, żeby na początku się orientować.

Finanse – kluczowe wskaźniki (KPI)

- **Metryki przychodów:** MRR, ARR, churn rate
- **Marże zysku:** brutto, operacyjna, netto
- **Przepływy pieniężne:** operacyjne, inwestycyjne, finansowe
- **Obliczenia ROI:** marketing, projekty
- **Budżet vs realizacja:** analiza odchyleń

Marketing i sprzedaż

- **Analiza lejka sprzedażowego:** współczynniki konwersji
- **Pozyskanie klienta:** CAC, LTV
- **Efektywność kampanii:** CTR, konwersje, ROAS
- **Pipeline sprzedażowy:** szanse, wskaźnik wygranych
- **Analiza kohort:** retencja, zachowania klientów

Operacje

- **Łańcuch dostaw:** rotacja zapasów, czasy realizacji
- **Metryki jakości:** wskaźniki defektów, satysfakcja klientów
- **Produktywność:** wskaźniki efektywności, wykorzystanie zdolności produkcyjnych
- **Poziomy usług:** SLA, czasy reakcji

Etap 4: Zaawansowane techniki analityczne

Analizy statystyczne w BI

- **Analiza trendów:** średnie ruchome, sezonowość
- **Prognozowanie:** podstawowe modele predykcyjne w Power BI/Tableau
- **Analiza regresji:** korelacja, zależność przyczynowa
- **Istotność statystyczna:** w testach A/B
- **Wykrywanie wartości odstających:** w kontekście biznesowym

Zaawansowane wizualizacje

- **Własne wizualizacje:** marketplace Power BI, rozszerzenia Tableau
- **Analizy geograficzne:** mapy, insighty geograficzne
- **Szeregi czasowe:** zaawansowane analizy temporalne
- **Scenariusze "co jeśli":** modelowanie parametryczne
- **Optymalizacja pod mobile:** responsywny design

Etap 5: Proces i metodyki

Agile BI

- **Zbieranie wymagań:** wywiady ze stakeholderami
- **Iteracyjny rozwój:** podejście MVP
- **Testy akceptacyjne użytkownika:** pętle feedbacku
- **Zarządzanie zmianą:** strategie adopcji użytkowników

Zarządzanie danymi (Data Governance)

- **Jakość danych:** walidacja, procesy czyszczenia
- **Master Data Management:** pojedyncze źródło prawdy
- **Bezpieczeństwo:** zabezpieczenia na poziomie wierszy, prywatność danych
- **Dokumentacja:** słownik danych, lineage
- **Kontrola wersji:** wersjonowanie raportów, wdrożeń

Etap 6: Umiejętności miękkie i komunikacja (ciągły rozwój)

Storytelling z danymi

- **Struktura narracji:** problem → rozwiązanie → rezultat
- **Hierarchia wizualna:** kierowanie uwagi odbiorcy
- **Teoria kolorów:** skuteczne użycie barw
- **Dostosowanie do odbiorcy:** techniczny vs biznesowy
- **Umiejętności prezentacji:** pokazy na żywo, obsługa Q&A

Współpraca biznesowa

- **Zarządzanie interesariuszami:** oczekiwania, priorytety
- **Tłumaczenie wymagań:** biznesowe na techniczne
- **Szkolenia użytkowników:** self-service analytics
- **Podejście konsultingowe:** doradca vs wykonawca

BI Analyst - typowe projekty

Podsumowanie projektów

- Zarządzanie strategiczne (Executive Dashboard)
- Sprzedaż (Sales Performance)
- Finanse (Financial Reporting)
- Marketing (Campaign Analyzer)
- Operacje (Operational Metrics)

Pokazują różne umiejętności techniczne

- **Integrację danych** z wielu źródeł
- **Real-time monitoring** vs raporty okresowe
- **Zaawansowane wizualizacje** (mapy, lejki, trendy)
- **Drill-down functionality** - od ogółu do szczegółów
- **Multi-channel attribution** - skomplikowana logika biznesowa

Demonstrują znajomość biznesu

- **KPI** specyficzne dla różnych działów
- **Customer journey** i punkty styku
- **Financial ratios** i wskaźniki
- **Operational excellence** metryki
- **Performance management** podejście

1. Executive Dashboard - Pulpit zarządu

Co to jest

Wysokopoziomowy dashboard dla zarządu firmy pokazujący najważniejsze wskaźniki biznesowe na jednym ekranie.

Kluczowe elementy

Finansowe KPI

- **Przychody** - miesięczne, kwartalne, roczne z trendem
- **Zysk operacyjny** - marża, porównanie do budżetu
- **Przepływ gotówki** - bieżąca sytuacja finansowa
- **ROI** - zwrot z inwestycji w kluczowe projekty

Operacyjne metryki

- **Sprzedaż** - wartość, liczba transakcji, średnia wartość zamówienia
- **Klienci** - liczba aktywnych, nowi klienci, wskaźnik odejść (churn)
- **Produktywność** - wydajność zespołów, realizacja celów
- **Jakość** - satysfakcja klientów, reklamacje

Funkcjonalność drill-down

- **Kliknięcie w przychody** → szczegóły po produktach/regionach
- **Kliknięcie w klientów** → analiza segmentów klientów
- **Kliknięcie w sprzedaż** → performance sprzedawców
- **Filtry czasowe** - dzień/tydzień/miesiąc/rok

Przykładowe źródła danych

- System ERP (SAP, Oracle)
- CRM (Salesforce, HubSpot)
- System finansowy
- Google Analytics (dla e-commerce)

Wyzwania techniczne

- **Agregacja danych** z różnych systemów
- **Aktualizacja w czasie rzeczywistym** lub codziennie rano
- **Bezpieczeństwo** - tylko wybrani mogą zobaczyć wszystkie dane
- **Mobilna responsywność** - zarząd ogląda na telefonach/tabletach

2. Sales Performance Tracker - Śledzenie wyników sprzedaży

Co to jest

Szczegółowa analiza wyników sprzedażowych dla menedżerów i przedstawicieli handlowych.

Kluczowe komponenty

Pipeline sprzedażowy

- **Lejek sprzedaży** - lead → prospect → opportunity → zamknięta sprzedaż
- **Conversion rates** - współczynniki konwersji między etapami
- **Średni czas w etapie** - jak długo trwa przejście lead→klient
- **Wartość pipeline'u** - potencjalne przychody w toku
- **Prognoza sprzedaży** - przewidywane zamknięcia w miesiącu/kwartale

Analiza terytorialna

- **Mapa sprzedaży** - wyniki po regionach/miastach
- **Top performing territories** - najlepsze regiony
- **Market penetration** - % rynku opanowanego w regionie
- **Konkurencja** - gdzie przegrywamy z konkurentami

Performance przedstawicieli

- **Ranking sprzedawców** - top performers vs bottom
- **Cel vs realizacja** - % wykonania planów sprzedażowych - w podziale na sprzedawców, ich zespoły czy regiony
- **Aktywność** - liczba spotkań, połączeń, emaili
- **Win rate** - % wygranych opportunities
- **Średnia wartość deala** - porównanie między sprzedawcami

Analiza produktowa

- **Bestsellery** - najlepiej sprzedające się produkty
- **Cross-selling** - które produkty sprzedawane razem
- **Marże** - rentowność różnych produktów
- **Sezonowość** - trendy sprzedażowe w roku

Przykładowe źródła danych

- CRM (Salesforce, Pipedrive, HubSpot)
- System ERP z danymi o produktach

- Dane geograficzne (mapy, demografia)
- Konkurencja (jeśli dostępne)

Funkcjonalności

- **Filtry** - okres, region, sprzedawca, produkt
- **Alerty** - gdy ktoś nie realizuje planu
- **Eksport do Excel** - dla szczegółowych analiz
- **Komentarze** - menedżerowie mogą dodawać notatki

3. Financial Reporting Suite - Pakiet raportów finansowych

Co to jest

Kompletny zestaw raportów finansowych dla CFO, księgowości i kontrolingu.

Główne raporty

Rachunek Zysków i Strat (P&L)

- **Przychody** - ze sprzedaży, z usług, inne przychody
- **Koszty bezpośrednie** - materiały, robocizna
- **Zysk brutto** - przychody minus koszty bezpośrednie
- **Koszty operacyjne** - marketing, administracja, sprzedaż
- **EBITDA** - zysk przed odsetkami, podatkami, amortyzacją
- **Zysk netto** - ostateczny wynik
- **Porównania** - rok do roku, budżet vs rzeczywistość

Bilans

- **Aktywa** - środki trwałe, zapasy, należności, gotówka
- **Pasywa** - kapitał własny, zobowiązania długo/krótkoterminowe
- **Wskaźniki** - płynność, zadłużenie, rotacja aktywów

Przepływ gotówki (Cash Flow)

- **Operacyjna** - pieniądze z podstawowej działalności
- **Inwestycyjna** - zakupy/sprzedaż majątku
- **Finansowa** - kredyty, dywidendy, emisje akcji
- **Prognoza** - przewidywany cash flow na kolejne miesiące

Budżet vs rzeczywistość

- **Variance analysis** - gdzie odbiegamy od planu
- **Rolling forecast** - aktualizowana prognoza na rok
- **Scenariusze** - optymistyczny/pesymistyczny/realistyczny

Zaawansowane analizy

- **Rentowność** - po produktach, klientach, projektach
- **Break-even analysis** - próg rentowności
- **Sensitivity analysis** - jak zmiany wpływają na wynik
- **Benchmarking** - porównanie z branżą

Źródła danych

- System księgowy (SAP, Oracle, Sage)
- Budżety z Excel/systemów planowania
- Dane operacyjne (sprzedaż, produkcja)
- Kursy walut (dla firm międzynarodowych)

4. Marketing Campaign Analyzer - Analiza kampanii marketingowych

Co to jest

Narzędzie do oceny skuteczności kampanii marketingowych i optymalizacji wydatków na marketing.

Multi-channel attribution (atrybucja wielokanałowa)

Źródła ruchu

- **Paid Search** - Google Ads, Bing Ads
- **Social Media** - Facebook, Instagram, LinkedIn, TikTok
- **Email Marketing** - newslettery, automatyzacje
- **Display Advertising** - bannery, retargeting
- **Organic** - SEO, direct traffic
- **Offline** - radio, TV, prasa, eventy

Customer Journey Mapping

- **Touchpoints** - wszystkie punkty kontaktu z marką
- **First-click attribution** - zasługa pierwszego kontaktu
- **Last-click attribution** - zasługa ostatniego kontaktu
- **Multi-touch attribution** - rozłożenie zasług proporcjonalnie
- **Time decay** - nowsze touchpoints mają większą wagę

Kluczowe metryki

Performance metryki:

- **CTR** (Click Through Rate) - % kliknięć w stosunku do wyświetleń
- **CPC** (Cost Per Click) - koszt za kliknięcie
- **CPM** (Cost Per Mille) - koszt za 1000 wyświetleń
- **CVR** (Conversion Rate) - % konwersji z odwiedzin **ROI metryki:**
- **ROAS** (Return on Ad Spend) - zwrot z wydatków reklamowych
- **CAC** (Customer Acquisition Cost) - koszt pozyskania klienta
- **LTV** (Lifetime Value) - wartość klienta przez cały cykl życia
- **Payback Period** - czas zwrotu inwestycji w klienta **Engagement metryki:**
- **Reach** - zasięg kampanii
- **Impressions** - liczba wyświetleń
- **Engagement Rate** - zaangażowanie (lajki, komentarze, udostępnienia)

- **Brand Awareness** - świadomość marki (badania/ankiety)

Funkcjonalności dashboardu

- **Porównanie kanałów** - który najskuteczniejszy
- **Optymalizacja budżetu** - realokacja środków
- **Cohort analysis** - jak różne kampanie wpływają długoterminowo
- **A/B testing** - porównanie wersji kampanii
- **Forecasting** - przewidywanie wyników

Źródła danych

- Google Analytics 4
- Facebook Ads Manager
- Google Ads
- Platformy email marketingowe (Mailchimp, GetResponse)
- CRM z danymi o konwersjach
- System e-commerce

5. Operational Metrics Dashboard - Dashboard metryk operacyjnych

Co to jest

Dashboard monitorujący kluczowe procesy operacyjne firmy w czasie rzeczywistym.

Obszary monitorowania

Produkcja/Manufacturing

- **OEE** (Overall Equipment Effectiveness) - efektywność maszyn
- **Downtime** - czas przestojów z przyczynami
- **Quality metrics** - % produktów wadliwych, reklamacje
- **Production rate** - sztuki/godzinę vs plan
- **Inventory levels** - poziomy zapasów surowców/produktów
- **Supply chain** - opóźnienia dostaw, lead times

Customer Service

- **Response time** - czas odpowiedzi na zapytania
- **Resolution time** - czas rozwiązania problemu
- **First Call Resolution** - % problemów rozwiązanych za pierwszym razem
- **Customer Satisfaction** - CSAT, NPS scores
- **Ticket volume** - liczba zgłoszeń dziennie/godzinnie
- **Agent performance** - produktywność konsultantów

Website/App Performance

- **Uptime** - dostępność serwisu (99.9%)
- **Page load time** - szybkość ładowania stron
- **Error rates** - 404, 500 i inne błędy
- **User experience** - bounce rate, session duration
- **Conversion funnel** - gdzie użytkownicy odpadają

Logistics/Fulfillment

- **Order processing time** - czas od zamówienia do wysyłki
- **Shipping performance** - % dostaw na czas
- **Return rate** - % zwrotów z przyczynami
- **Last mile delivery** - performance kurierów

Real-time monitoring

- **Live data feeds** - aktualizacja co minutę/5 minut
- **Alerts i notifications** - SMS/email gdy coś pójdzie nie tak
- **Traffic light system** - zielone/żółte/czerwone wskaźniki
- **Trend analysis** - czy metryki się pogarszają
- **Historical comparison** - dzisiaj vs wczoraj/tydzień temu

Przykładowe alerty

- “Czas odpowiedzi przekroczył 2 minuty”
- “Maszyna #3 nie działa od 15 minut”
- “Zapasy produktu X poniżej minimum”
- “Conversion rate spadł o 20% vs wczoraj”

Źródła danych

- Systemy produkcyjne (MES, SCADA)
- Customer service platforms (Zendesk, Freshdesk)
- Web analytics (Google Analytics, Adobe)
- Monitoring aplikacji (New Relic, Datadog)
- WMS (Warehouse Management System)
- ERP systemy

BI Analyst - aspekty techniczne

Narzędzia do tworzenia

- **Power BI** - działa dobrze z innymi produktami Microsoft, DAX, Power Query
- **Tableau** - drag & drop, zaawansowane wizualizacje
- **Qlik Sense** - narzędzie które automatycznie znajduje powiązania między danymi i pozwala użytkownikom samodzielnie tworzyć analizy
- **Looker/Google Data Studio** - narzędzia Google działające w chmurze, oparte na języku zapytań SQL

Architektura danych

- **Źródła danych** - różne systemy z których pobieramy dane (np. systemy sprzedażowe, księgowe)
- **ETL/ELT** - proces pobierania, czyszczenia i przygotowywania danych do analizy
- **Hurtownia danych** - centralne miejsce gdzie są składowane wszystkie czyste dane
- **Warstwa raportowania** - gdzie tworzone są wykresy, dashboardy i raporty
- **Użytkownicy** - różne grupy ludzi (menedżerowie, analitycy) z różnymi poziomami dostępu

Dobre praktyki

- **Wydajność** - raporty muszą ładować się szybko, nawet z dużą ilością danych
- **Bezpieczeństwo** - każdy widzi tylko te dane, które powinien widzieć
- **Zarządzanie danymi** - każda liczba/wskaźnik ma jedno, oficjalne źródło prawdy
- **Wygoda użytkowania** - raporty muszą być intuicyjne i działać na telefonach
- **Dokumentacja** - jasne opisy co oznaczają poszczególne wskaźniki

Umiejętności potrzebne do stworzenia

- **SQL** - zaawansowane zapytania, window functions
- **Narzędzia BI** - Power BI/Tableau na średnim poziomie
- **Modelowanie danych** - umiejętność projektowania struktury danych (jak połączyć tabele)
- **Znajomość biznesu** - rozumienie jak działa firma i jakie ma procesy
- **Myślenie projektowe** - umiejętność przewidywania jak użytkownicy będą korzystać z raportów

BI Analyst - ćwiczenia

Mini-projekt 1: Dashboard sprzedaży dla małego sklepu

- **Cel:** Stwórz interaktywny dashboard w Power BI lub Tableau (wybierz jedno narzędzie), który wizualizuje kluczowe wyniki sprzedaży.
- **Kroki:** Znajdź publiczny zestaw danych o sprzedaży (np. Superstore na Kaggle). Wczytaj dane do wybranego narzędzia BI. Zbuduj **wykresy trendów sprzedaży**, **sprzedaż według kategorii/produktów**, **mapę sprzedaży** (jeśli dane zawierają geolokalizację). Zaimplementuj **filtry czasowe** (np. dzień/tydzień/miesiąc/rok) i **funkcjonalność drill-down** (np. z sumy sprzedaży do poszczególnych transakcji lub do poszczególnych sprzedawców).
- **Pobudzanie wyobraźni:** Jakie **KPI** sprzedaży są kluczowe dla zarządu (np. przychody, liczba transakcji, średnia wartość zamówienia)? Jakie inne źródła danych (np. marketingowe, magazynowe) można by zintegrować, aby uzyskać pełniejszy obraz działania sklepu?

Mini-projekt 2: Monitorowanie kampanii marketingowej

- **Cel:** Zbuduj dashboard oceniający skuteczność kampanii marketingowej, aby pomóc działowi marketingu w optymalizacji wydatków.
- **Kroki:** Wyobraź sobie lub znajdź uproszczone dane z kampanii (np. wydatki na reklamy, kliknięcia, konwersje, wyświetlenia). Oblicz kluczowe metryki takie jak **CTR (Click Through Rate), CPC (Cost Per Click), CVR (Conversion Rate), ROAS (Return on Ad Spend)**. Wizualizuj te metryki w wybranym narzędziu BI. Zastanów się, jak zastosować **multi-channel attribution** (jak przypisać zasługę za konwersję różnym kanałom marketingowym).
- **Pobudzanie wyobraźni:** Jakie metryki są najważniejsze dla działu marketingu i dlaczego? Jak ten dashboard mógłby pomóc w alokacji budżetu reklamowego między różnymi kanałami?

Mini-projekt 3: Rachunek Zysków i Strat (P&L) dla fikcyjnej firmy

- **Cel:** Stwórz raport finansowy Rachunek Zysków i Strat (P&L) dla zarządu lub działu finansowego.
- **Kroki:** Wymyśl lub znajdź uproszczone dane finansowe (przychody ze sprzedaży, koszty bezpośrednie, koszty operacyjne, koszty sprzedaży). Zbuduj tabelę P&L w narzędziu BI, obliczając **zysk brutto, EBITDA, zysk netto**. Dodaj porównanie **rok do roku** lub do budżetu.

- **Pobudzanie wyobraźni:** Jakie inne raporty finansowe są ważne dla firm (np. bilans, przepływ gotówki)? Jakie wskaźniki rentowności są kluczowe dla CFO? Jak można by wykorzystać ten raport do analizy odchyleń od budżetu?

Mini-projekt 4: Dashboard metryk operacyjnych

- **Cel:** Stwórz dashboard monitorujący kluczowe procesy operacyjne firmy w czasie rzeczywistym (lub z symulacją danych aktualizowanych często).
- **Kroki:** Wymyśl dane o procesach operacyjnych, np. z działu obsługi klienta (czas odpowiedzi, czas rozwiązania problemu, ocena satysfakcji), lub produkcji (liczba wyprodukowanych sztuk, czas przestoju maszyny). Oblicz **średni czas odpowiedzi**, **liczbę zgłoszeń dziennie**, lub **efektywność maszyn**. Stwórz wykresy trendów dla tych metryk. Zaimplementuj symulowane alerty (np. wskaźnik zmieni kolor na czerwony, gdy wartość przekroczy próg).
- **Pobudzanie wyobraźni:** Jakie alerty mogłyby być wysyłane, gdy czas odpowiedzi przekroczy próg? Jakie inne źródła danych (np. logi z czatów, dane z czujników) można by wykorzystać do głębszej analizy procesów operacyjnych?

BI Analyst - quiz

Etap 1: Zaawansowane narzędzia BI

Pytanie 1: Wymień dwa popularne narzędzia klasy Business Intelligence (BI) wspomniane w źródłach.

Odpowiedź 1: Przykładami popularnych narzędzi BI są Power BI, Tableau, Qlik Sense czy Looker/Google Data Studio.

Pytanie 2: W narzędziu BI, do czego służą parameters i filters?

Odpowiedź 2: Parameters i filters służą do dodawania interaktywności do dashboardów, pozwalając użytkownikom dynamicznie zmieniać widok danych.

Etap 2: Modelowanie danych dla BI

Pytanie 1: Czym charakteryzuje się schemat gwiazdy (star schema) w modelowaniu hurtowni danych?

Odpowiedź 1: Schemat gwiazdy składa się z centralnej tabeli faktów otoczonej przez powiązane z nią bezpośrednio tabele wymiarów.

Pytanie 2: Do czego służą Aggregation tables w kontekście optymalizacji hurtowni danych dla BI?

Odpowiedź 2: Aggregation tables to tabele zawierające wstępnie obliczone podsumowania danych z tabel faktów, co przyspiesza wykonywanie zapytań dla raportów.

Etap 3: Biznesowa domena wiedzy

Pytanie 1: W kontekście finansowych KPI, co oznacza MRR i ARR?

Odpowiedź 1: MRR (Monthly Recurring Revenue) to miesięczne cykliczne przychody, a ARR (Annual Recurring Revenue) to roczne cykliczne przychody – kluczowe metryki dla modeli subskrypcyjnych.

Pytanie 2: W kontekście marketingu i sprzedaży, co oznacza CAC i LTV?

Odpowiedź 2: CAC (Customer Acquisition Cost) to koszt pozyskania klienta, a LTV (Lifetime Value) to wartość klienta przez cały okres jego relacji z firmą.

Etap 4: Zaawansowane analizy i wizualizacje (Bazując na opisach projektów)

Pytanie 1: Jaka technika analityczna pozwala śledzić, ilu użytkowników przechodzi przez poszczególne etapy procesu (np. od odwiedzenia strony do zakupu)?

Odpowiedź 1: Służy do tego analiza lejka (funnel analysis).

Pytanie 2: Co to jest multi-channel attribution w analizie kampanii marketingowych?

Odpowiedź 2: Multi-channel attribution to przypisywanie zasługi za konwersję (np. zakup) poszczególnym punktom kontaktu (touchpoints) klienta z marką na różnych kanałach marketingowych podczas jego ścieżki zakupowej.

Etap 5: Storytelling z danymi i komunikacja

Pytanie 1: Dlaczego storytelling z danymi jest ważną umiejętnością dla BI Analityka?

Odpowiedź 1: Storytelling z danymi pozwala na efektywne komunikowanie złożonych wniosków z analiz w sposób przekonujący i zrozumiały dla odbiorców, zwłaszcza biznesowych.

Pytanie 2: W kontekście wizualizacji danych, co oznacza Visual hierarchy i Color theory?

Odpowiedź 2: Visual hierarchy to zasada projektowania dashboardów, która prowadzi uwagę użytkownika przez najważniejsze elementy. Color theory to zasady efektywnego używania kolorów w wizualizacjach, aby podkreślić kluczowe informacje i zapewnić czytelność.

Etap 6: Umiejętności miękkie i współpraca z biznesem

Pytanie 1: Wymień dwie kluczowe umiejętności miękkie, które są niezbędne do sukcesu w zawodach związanych z danymi.

Odpowiedź 1: Kluczowe umiejętności to m.in. rozwiązywanie problemów, komunikacja, umiejętności analityczne i umiejętności badawcze.

Pytanie 2: Na czym polega umiejętność Requirements translation (tłumaczenie wymagań) w pracy z biznesem?

Odpowiedź 2: Polega na zdolności do przekładania niejasnych potrzeb i pytań biznesowych na konkretne, mierzalne problemy analityczne lub techniczne, które można rozwiązać za pomocą danych.

Kluczowe umiejętności interpersonalne

Obok umiejętności technicznych, umiejętności miękkie są niezbędne do sukcesu w zawodach związanych z danymi. Najważniejsze z nich to:

Rozwiązywanie problemów

Wykwalifikowany analityk danych nie tylko rozwiązuje zdefiniowane problemy, ale także proaktywnie identyfikuje i definiuje nowe. Umiejętność radzenia sobie z niepewnością i brakiem z góry określonych kroków jest kluczowym elementem rozwiązywania problemów w data science.

Komunikacja

Efektywna komunikacja jest niezbędna w dziedzinie data science, ponieważ umożliwia wymianę pomysłów, nauki i spostrzeżeń w ramach zespołu i pomiędzy osobami. Umiejętność jasnego wyrażania swoich poglądów, skutecznej współpracy i osiągania znaczących wyników jest fundamentalna dla analityków danych.

Umiejętności analityczne

Obejmują zdolność do dekonstrukcji informacji, analizy danych i wyciągania wniosków poprzez logiczne rozumowanie, krytyczne myślenie, komunikację, badania i kreatywność. Te umiejętności są niezbędne do rozwiązywania problemów i podejmowania decyzji.

Umiejętności badawcze

Odnoszą się do zdolności i technik stosowanych w procesie zdobywania i rozumienia nowych informacji, zachowywania wiedzy i przeprowadzania ocen. Obejmują różne metody, takie jak efektywne czytanie, techniki koncentracji i efektywne sporządzanie notatek.

Zarządzanie projektem i współpracą w zespole

Wykorzystanie narzędzi do zarządzania projektami (np. Jira, Trello), współpraca z innymi specjalistami (np. product managerami, developerami, ekspertami dziedzinowymi) i zrozumienie ich (oraz własnej) ról w zespole/projekcie. W ramach wdrażania rozwiązań - stosowanie zasad CI/CD, repozytorium kodu, jego wersjonowania czy też stylu pisania dokumentacji.

Rozwijanie wiedzy domenowej

Wiedza domenowa dla analityków danych odnosi się do dziedziny, w której znajdują się analizowane dane. Czasami domena danych może dostarczyć użytecznych spostrzeżeń i pomóc analitykom łatwiej rozszyfrować i zrozumieć dane.

Zdobycie wiedzy w konkretnej branży (finanse, opieka zdrowotna, e-commerce, itp.) może znacząco zwiększyć Twoją wartość jako analityka danych i pomóc w wyciąganiu bardziej wartościowych wniosków z analizowanych danych.

Trudno podać konkretny plan nauki czy też zdobywania doświadczenia. Ale przygotowałem kilka propozycji kluczowych zagadnień dla poszczególnych ról w różnych branżach (a nawet dość szczegółowych fragmentach/zagadnieniach z tych branż).

Analityk danych/BI w e-commerce i marketingu

E-commerce (sklepy internetowe)

- **Lejek sprzedażowy** - analiza tego, jak klienci przechodzą przez kolejne etapy: od wejścia na stronę, przez przeglądanie produktów, dodanie do koszyka, aż po finalizację zakupu
- **Współczynnik konwersji** - jaki procent odwiedzających faktycznie coś kupuje
- **Wartość życiowa klienta (CLV)** - ile pieniędzy przyniesie nam przeciętny klient przez cały okres współpracy
- **Koszt pozyskania klienta (CAC)** - ile kosztuje nas zdobycie jednego nowego klienta
- **Analiza koszykowa** - co klienci kupują razem, jakie produkty się uzupełniają
- **Testowanie A/B** - porównywanie dwóch wersji strony/reklamy, żeby zobaczyć która działa lepiej
- **Sezonowość** - jak sprzedaż zmienia się w czasie (święta, wakacje, Black Friday)
- **Zwrot z inwestycji reklamowej (ROAS)** - ile zarabiamy na każdej złotówce wydanej na reklamę

Marketing cyfrowy

- **Modele atrybucji** - jak przypisać zasługę za sprzedaż różnym kanałom marketingowym (czy to reklama na Facebook, Google, czy email był decydujący?)
- **Marketing w wyszukiwarkach (SEO/SEM)** - jak ludzie nas znajdują w Google i ile to kosztuje
- **Analiza mediów społecznościowych** - jak nasze posty działają na Facebooku, Instagramie
- **Email marketing** - ile osób otwiera nasze maile, klika w linki

- **Mapa podróży klienta** - przez jakie kanały przechodzi klient zanim kupi

Data Scientist - wykrywanie oszustw w bankach

Branża bankowa

- **Regulacje prawne** - RODO (ochrona danych), przepisy antypraniowe, wymogi nadzoru finansowego
- **Weryfikacja tożsamości klienta (KYC)** - jak bank sprawdza, że klient to rzeczywiście ten za kogo się podaje
- **Ocena zdolności kredytowej** - jak bank decyduje, czy dać komuś kredyt
- **Systemy płatnicze** - jak działają przelewy, płatności kartą, BLIK. Ciekawostka: moje ulubione pytanie podczas rekrutacji to *jak działa bankomat?*

Wykrywanie oszustw

- **Rodzaje oszustw** - kradzież kart, przejęcie konta, fałszywe tożsamości, pranie pieniędzy
- **Analiza w czasie rzeczywistym** - system musi decydować o transakcji w ułamku sekundy
- **Równowaga między bezpieczeństwem a wygodą** - jak nie blokować normalnych transakcji, ale wykryć podejrzane
- **Analiza zachowań** - rozpoznawanie nietypowych wzorców (np. ktoś nagle zaczął robić zakupy w innym mieście)
- **Analiza powiązań** - jak oszuści mogą być ze sobą powiązani przez adresy, telefony, karty
- **Wyjaśnialność algorytmów** - bank musi umieć wytłumaczyć klientowi, dlaczego zablokował transakcję

Inżynier danych w portalu aukcyjnym

Architektura systemów aukcyjnych

- **Przetwarzanie w czasie rzeczywistym** - system musi natychmiast reagować na każdą licytację
- **Obsługa dużego ruchu** - szczególnie podczas popularnych aukcji, gdy tysiące osób licytuje jednocześnie
- **Architektura oparta na zdarzeniach** - każda licytacja, każde kliknięcie generuje dane, które trzeba przetworzyć
- **Synchronizacja danych** - jak zapewnić, że wszyscy widzą aktualną cenę w tym samym momencie

Specyfika danych aukcyjnych

- **Dane szeregów czasowych** - historia wszystkich licytacji, minuta po minucie
- **Śledzenie zachowań użytkowników** - kto, kiedy i jak licytuje, jakie ma wzorce
- **Zarządzanie magazynem** - ile produktów jest dostępnych, co się sprzedaje
- **Procesy płatności** - jak bezpiecznie i szybko przetworzyć płatność po wygranej aukcji
- **Systemy rekomendacji** - jakie aukcje mogą zainteresować konkretnego użytkownika
- **Wyszukiwanie produktów** - jak ludzie szukają tego, co chcą kupić
- **Różnice między urządzeniami** - jak inaczej zachowują się użytkownicy na telefonie i komputerze

Infrastruktura techniczna

- **Przetwarzanie strumieni danych** - ciągłe analizowanie napływających danych
- **Pamięć podręczna** - jak szybko wyświetlać często oglądane produkty
- **Podział baz danych** - jak przechowywać ogromne ilości danych o aukcjach
- **Monitoring systemów** - jak szybko wykryć, że coś nie działa
- **Jakość danych w czasie rzeczywistym** - jak upewnić się, że dane są poprawne
- **Kopie zapasowe** - jak nie stracić danych o transakcjach warte miliony

Każda z tych ról wymaga nie tylko umiejętności technicznych, ale przede wszystkim zrozumienia biznesu i branży, w której pracuje.

Nie da się rozumieć wszystkich branż, w szczególności bardzo detalicznie. Zdecydowanie się na jedną być może zamknie Cię (na jakiś czas) w jednym temacie. To nie jest problem, o ile temat Cię interesuje. Warto jednak szukać poza określoną domenę biznesową i porównywać istniejące problemy - i przede wszystkim rozwiązania - pomiędzy różnymi biznesami. Koniec końców wiele rozwiązań sprawdza się w różnych dziedzinach!

Etyka danych i prywatność

Do tej pory mówiliśmy o tym co trzeba/warto umieć, jakie narzędzia opanować, jakie projekty zrobić. Ale praca z danymi wiąże się z zagadnieniami etycznymi. Każda z ról może mieć przypisane dla siebie zagadnienia, wszyscy jednak powinniśmy *używać danych* tak, aby nikogo nie krzywdzić lub nie naruszać określonych zasad.

Jakie mamy wyzwania?

Analityk danych/BI

Główne zagadnienia etyczne

- **Anonimizacja vs użyteczność danych** - jak chronić tożsamość klientów, zachowując wartość analityczną danych
- **Reprezentatywność próby** - unikanie stronniczości w raportach przez niewłaściwy dobór danych
- **Transparentność metodologii** - jasne komunikowanie ograniczeń i założeń analiz
- **Zgodność z RODO** - prawidłowe przetwarzanie danych osobowych w analizach

Przykłady dylematów

Przykład 1: Analiza segmentacji klientów Analityk tworzy segmenty klientów na podstawie danych demograficznych i zakupowych. Jeden z segmentów wyraźnie koreluje z pochodzeniem etnicznym. Czy można go wykorzystać w strategii marketingowej? Ryzyko: dyskryminacja cenowa lub wykluczenie grup społecznych.

Przykład 2: Raportowanie wyników A/B testów Test pokazuje, że nowa wersja strony zwiększa sprzedaż o 5%, ale głównie kosztem starszych użytkowników, którzy mają problemy z nawigacją. Czy raportować tylko ogólny wynik, czy też wspomnieć o tym problemie?

Przykład 3: Analiza produktywności pracowników HR prosi o analizę danych z systemów firmowych, aby zidentyfikować najmniej produktywnych pracowników. Analityk ma dostęp do danych o logowaniach, emailach, obecności. Jak zrównoważyć potrzeby biznesowe z prywatnością pracowników?

Data Scientist

Główne zagadnienia etyczne

- **Stronniczość algorytmów (bias)** - jak algorytmy mogą dyskryminować różne grupy

- **Wyjaśnialność modeli** -(*explainable AI*, *XAI*) czy użytkownicy rozumieją, jak podejmowane są decyzje
- **Wpływ na społeczeństwo** - długoterminowe konsekwencje wdrożenia modeli
- **Zgodność z regulacjami** - szczególnie w sektorach regulowanych (finanse, medycyna)

Przykłady dylematów

Przykład 1: Model oceny kredytowej Model uczony na historycznych danych pokazuje, że osoby z pewnych dzielnic mają wyższe ryzyko niewypłacalności. Czy uwzględnić kod pocztowy w modelu? Ryzyko: utrwalanie nierówności społecznych i geograficznych. Podobny - często opisywany - problem dotyczy przestępczości.

Przykład 2: System rekomendacji treści Algorytm doskonale przewiduje, jakie treści będą angażować użytkowników, ale preferuje kontrowersyjne i polaryzujące materiały. Czy optymalizować engagement czy dobro społeczne?

Przykład 3: Wykrywanie oszustw w ubezpieczeniach Model identyfikuje podejrzane zgłoszenia szkód, ale częściej "podejrzewa" osoby o niższych dochodach. Jak zapewnić sprawiedliwość, nie tracąc skuteczności?

Przykład 4: Predykcja zachowań konsumenckich Model potrafi przewidzieć ciążę na podstawie zakupów (słynny przypadek Target). Czy można używać takich predykcji w marketingu, jeśli klientka jeszcze nie wie, że jest w ciąży?

Inżynier danych

Główne zagadnienia etyczne

- **Bezpieczeństwo danych** - ochrona przed wyciekami i nieautoryzowanym dostępem
- **Minimalizacja zbierania danych** - czy zbieramy tylko to, co naprawdę potrzebne
- **Prawo do zapomnienia** - jak skutecznie usuwać dane na żądanie użytkowników
- **Kontrola dostępu** - kto, kiedy i do jakich danych ma dostęp

Przykłady dylematów

Przykład 1: Architektura data lake'a Firma chce zbierać "wszystkie możliwe dane, może się przydadzą w przyszłości". Inżynier ma zaprojektować system, który będzie gromadził maksymalnie dużo informacji o użytkownikach. Jak zrównoważyć potrzeby biznesowe z minimalizacją danych?

Przykład 2: Logowanie działań użytkowników System loguje każde kliknięcie, każdy ruch myszy, czas spędzony na stronie. Te dane pomagają w optymalizacji, ale tworzą bardzo szczegółowy profil zachowań. Jak długo przechowywać takie dane? Czy informować użytkowników o takim poziomie śledzenia?

Przykład 3: Udostępnianie danych partnerom Firma ma umowy z partnerami biznesowymi na udostępnianie danych klientów. Inżynier ma zaprojektować API, które będzie te dane przekazywać. Jak zapewnić, że partnerzy dostaną tylko to, na co mają zgodę klientów?

Przykład 4: Backup i archiwizacja Klient żąda usunięcia swoich danych (prawo do zapomnienia), ale dane znajdują się w backupach z ostatnich 7 lat, w różnych systemach, w różnych formatach. Ile czasu i środków przeznaczyć na pełne usunięcie vs praktyczne minimum?

Przykład 5: Monitoring wydajności vs prywatność System monitorujący wydajność aplikacji zbiera informacje o błędach, ale często zawierają one dane osobowe użytkowników (emaile, ID, fragmenty wiadomości). Jak monitorować system, nie naruszając prywatności?

Wspólne wyzwania dla wszystkich ról

- **Świadomość konsekwencji:** Jak nasze decyzje techniczne wpływają na prawdziwych ludzi?
- **Balans między innowacją a ochroną:** Jak nie zatrzymać rozwoju, chroniąc prywatność?
- **Edukacja biznesu:** Jak wyjaśnić interesariuszom, dlaczego etyka danych to nie tylko “nice to have”?
- **Międzynarodowe różnice:** Jak radzić sobie z różnymi standardami prywatności w różnych krajach?

Kluczowe jest, aby każda z ról rozumiała, że praca z danymi to nie tylko zagadnienie techniczne, ale też społeczne i etyczne. Decyzje podejmowane na poziomie kodu czy architektury mają realny wpływ na życie milionów ludzi.

AI - pomoc czy przeszkoda?

Sztuczna inteligencja fundamentalnie zmienia krajobraz pracy z danymi, stawiając przed analitykami, data scientistami i inżynierami danych pytanie: **czy AI to potężne narzędzie wspomagające, czy też zagrożenie dla ich przyszłości zawodowej?** Odpowiedź, jak to często bywa, leży gdzieś pośrodku i zależy od tego, jak szybko i skutecznie specjaliści zaadoptują nowe technologie do swojej codziennej pracy. Tak, **to zależy**, jak zawsze w IT.

Modele językowe takie jak GPT, Claude czy Gemini rewolucjonizują sposób, w jaki pracujemy z kodem i dokumentacją. Analityk może teraz poprosić AI o napisanie zapytania SQL, wygenerowanie komentarzy do złożonych analiz czy nawet o interpretację wyników statystycznych. Data scientist otrzymuje pomoc w pisaniu kodu Pythona do modelowania, a inżynier danych może automatyzować tworzenie pipeline'ów ETL. To wszystko oznacza, że rutynowe, powtarzalne zadania mogą być wykonywane znacznie szybciej, uwalniając czas na bardziej strategiczne myślenie.

Automatyzacja procesów to kolejny obszar, gdzie AI pokazuje swoją siłę. Narzędzia takie jak AutoML potrafią samodzielnie wybierać najlepsze algorytmy, optymalizować hiperparametry czy nawet przeprowadzać feature engineering. Systemy automatycznego raportowania mogą generować codzienne dashboardy bez ludzkiej interwencji (ale traktowanie tego zdania dosłownie to gruba przesada!), a inteligentne alerty (...wcześniej zdefiniowane lub nauczone w procesach ML) potrafią wykryć anomalie w danych i powiadomić odpowiednie osoby. Dla niektórych może to oznaczać strach o miejsce pracy, dla innych - możliwość skupienia się na bardziej kreatywnych aspektach swojej roli.

Kluczowa zmiana polega na **przesunięciu akcentów w umiejętnościach**. O ile wcześniej dużą część czasu zajmowało mechaniczne pisanie kodu czy przygotowywanie raportów, teraz coraz większą wartość ma umiejętność zadawania właściwych pytań, interpretowania wyników i przekładania technicznej wiedzy na język biznesowy. **AI jest** (już nie *staje się*!) **potężnym asystentem**, ale wciąż potrzebuje ludzkiego nadzoru, kreatywności i zrozumienia kontekstu biznesowego.

Paradoksalnie, rozwój AI może nawet zwiększyć zapotrzebowanie na specjalistów od danych. Im więcej organizacji adoptuje rozwiązania AI, tym większa potrzeba na osoby, które potrafią te systemy wdrażać, monitorować i optymalizować. Zmienia się jednak profil idealnego kandydata - oprócz umiejętności technicznych coraz ważniejsze stają się soft skills, umiejętność współpracy z AI oraz zdolność do ciągłego uczenia się nowych narzędzi.

Ostatecznie, AI w kontekście ról związanych z danymi to nie rewolucja eliminująca ludzi, ale ewolucja współpracy człowiek-maszyna. Ci, którzy potrafią wykorzystać AI jako narzędzie do amplifikacji swoich umiejętności, będą mieli przewagę konkurencyjną. Ci, którzy będą

ignorować te zmiany lub postrzegać AI wyłącznie jako zagrożenie, mogą zostać w tyle. Przyszłość należy do hybrydowych zespołów, gdzie ludzka intuicja, kreatywność i zrozumienie biznesu łączą się z obliczeniową mocą i szybkością sztucznej inteligencji.

A gdzie konkretnie można wykorzystać AI? Stan wiedzy na połowę 2025 roku ;-)

Data Scientist

Wsparcie znaczące (trudniej do pełnej automatyzacji)

- Automatyczne feature engineering i selekcja zmiennych
- Sugerowanie odpowiednich algorytmów ML dla konkretnych problemów
- Hyperparameter tuning i optymalizacja modeli
- Generowanie kodu do eksploracyjnej analizy danych (EDA)
- Automatyczne tworzenie dokumentacji modeli

Data Engineer

Automatyzacja kompletna

- Generowanie podstawowych skryptów ETL/ELT na podstawie opisu wymagań
- Automatyczne wykrywanie i naprawianie prostych błędów w pipeline'ach danych
- Generowanie kodu do połączeń z bazami danych i API
- Tworzenie podstawowych testów jednostkowych dla pipeline'ów

Wsparcie znaczące

- Optymalizacja zapytań SQL przez analizę planów wykonania
- Sugerowanie najlepszych praktyk architektonicznych
- Monitorowanie i alertowanie o anomaliach w przepływie danych
- Automatyczne skalowanie infrastruktury w chmurze

Business Intelligence Analyst

Automatyzacja kompletna

- Generowanie standardowych raportów i dashboardów
- Automatyczne aktualizowanie wizualizacji na podstawie nowych danych
- Tworzenie podstawowych alertów biznesowych
- Generowanie opisów trendów i wzorców w danych

Wsparcie znaczące

- Sugerowanie najlepszych typów wizualizacji dla konkretnych danych

- Automatyczne wykrywanie anomalii w KPI
- Generowanie insights i rekomendacji biznesowych
- Tworzenie narracji do prezentacji danych (storytelling)

Elementy pracy które mogą być zautomatyzowane

Zadania rutynowe i powtarzalne

- **Czyszczenie danych:** usuwanie duplikatów, wypełnianie braków, standaryzacja formatów
- **Podstawowe analizy statystyczne:** obliczanie miar tendencji centralnej, korelacji
- **Generowanie raportów:** standardowe raporty miesięczne/kwartalne
- **Walidacja danych:** sprawdzanie poprawności, kompletności i spójności

Tworzenie kodu

- Podstawowe zapytania SQL na podstawie opisu w języku naturalnym
- Skrypty Python do manipulacji danych (pandas, numpy)
- Kod do tworzenia standardowych wizualizacji
- Automatyzacja połączeń z różnymi źródłami danych

Dokumentacja i komunikacja

- Automatyczne generowanie dokumentacji technicznej
- Tworzenie opisów metodologii i wyników analiz
- Przygotowywanie prezentacji z wynikami analiz
- Tłumaczenie wyników technicznych na język biznesowy

Obszary gdzie człowiek pozostaje niezbędny

Myślenie strategiczne i kontekstowe

- Interpretacja wyników w kontekście biznesowym
- Podejmowanie decyzji o kierunku analiz
- Zrozumienie niuansów branżowych i organizacyjnych
- Definiowanie problemów biznesowych do rozwiązania

Kreatywność i innowacyjność

- Projektowanie niestandardowych rozwiązań analitycznych
- Tworzenie nowych metod analizy specyficznych dla domeny

- Eksperymentowanie z nowatorskimi podejściami
- Adaptacja rozwiązań do unikalnych potrzeb organizacji

Relacje międzyludzkie

- Współpraca z interesariuszami biznesowymi
- Prezentowanie wyników i przekonywanie do rekomendacji
- Zarządzanie projektami i zespołami
- Rozwiązywanie konfliktów i negocjacje

Odpowiedzialność i etyka

- Zapewnienie zgodności z regulacjami (RODO, branżowe)
- Podejmowanie decyzji etycznych dotyczących wykorzystania danych
- Odpowiedzialność za wpływ analiz na biznes i społeczeństwo
- Kontrola jakości i wiarygodności wyników

Praktyczne rekomendacje dla specjalistów

Jak przygotować się na zmiany

- **Rozwijaj umiejętności prompt engineering** - naucz się efektywnie komunikować z AI
- **Skup się na umiejętnościach miękkich** - komunikacja, myślenie krytyczne, rozwiązywanie problemów
- **Pogłębiaj wiedzę domenową** - znajomość branży i kontekstu biznesowego
- **Ucz się współpracy z AI** - traktuj AI jako narzędzie wspomagające, nie konkurenta

Nowe role i możliwości

- **AI/ML Engineer** - specjalizacja w implementacji rozwiązań AI
- **Prompt Engineer** - optymalizacja komunikacji z systemami AI
- **AI Ethics Specialist** - zapewnienie etycznego wykorzystania AI
- **Human-AI Collaboration Specialist** - projektowanie procesów łączących ludzi i AI

AI będzie coraz bardziej wspomagać pracę w dziedzinie danych, ale kluczowe pozostaną umiejętności analitycznego myślenia, zrozumienia kontekstu biznesowego i komunikacji z interesariuszami. Specjaliści, którzy nauczą się efektywnie współpracować z AI, będą mieli przewagę konkurencyjną na rynku pracy.

Pozostałe umiejętności - ćwiczenia

Umiejętności Interpersonalne

Ćwiczenie 1: Rozwiązywanie problemów – Case Study

Wybierz złożony problem z życia codziennego (np. dlaczego autobusy są często spóźnione? Dlaczego ludzie tracą motywację do ćwiczeń?). Spróbuj go zdefiniować, zidentyfikować możliwe przyczyny (nawet bez danych) i zaproponować rozwiązania. Skup się na **radzeniu sobie z niepewnością i brakiem z góry określonych kroków**.

Ćwiczenie 2: Komunikacja – Prezentacja "dla laika"

Wybierz jeden z projektów, które wykonałeś (np. wizualizację w Excelu lub Power BI) i przygotuj krótką, 5-minutową prezentację dla osoby, która nie zna się na analizie danych (np. dla "zarządu" lub "działu marketingu"). **Skup się na wnioskach i ich znaczeniu biznesowym**, a nie na technicznych detalach czy kodzie.

Ćwiczenie 3: Współpraca w zespole – Fikcyjny projekt

Wyobraź sobie, że pracujesz w zespole danych nad projektem. Zastanów się, jakbyś dzielił zadania, komunikował postępy (np. przez codzienne "standupy"), rozwiązywał konflikty z kolegami i dbał o **jakość kodu i dokumentację**. Użyj narzędzi typu Trello/Jira do zorganizowania fikcyjnych zadań.

Wiedza Domenowa

Ćwiczenie 1: Zbadaj wybraną branżę

Wybierz branżę, która Cię interesuje (np. finanse, opieka zdrowotna, e-commerce, logistyka). Zbadaj kluczowe trendy, specyficzne problemy i terminologię używaną w tej branży. Spróbuj znaleźć przykładowe dane (np. raporty branżowe, otwarte dane publiczne).

Ćwiczenie 2: Tłumaczenie potrzeb biznesowych

Wyobraź sobie, że menedżer prosi Cię o "poprawę wyników sprzedaży" lub "zmniejszenie kosztów operacyjnych". Jakie pytania byś zadał, aby przetłumaczyć tę niejasną prośbę na konkretny, mierzalny problem analityczny lub techniczny? Co byś chciał wiedzieć o biznesie, aby dobrze go zrozumieć i sformułować problem?

Etyka Danych i AI

Ćwiczenie 1: Dylemat etyczny

Rozważ scenariusz, w którym masz dostęp do wrażliwych danych klientów (np. danych medycznych lub finansowych). Zastanów się, jakie są potencjalne ryzyka ich wykorzystania i jak zapewnić **prywatność danych** (np. poprzez anonimizację) oraz zgodność z regulacjami (np. **RODO/GDPR**).

Ćwiczenie 2: Współpraca z AI

Użyj narzędzia AI (np. ChatGPT, Claude) do pomocy w debugowaniu kodu Pythona, napisaniu zapytania SQL, czy generowaniu pomysłów na wizualizacje. **Poćwicz prompt engineering** – jak precyzyjnie formułować zapytania, aby uzyskać najlepsze odpowiedzi. Zastanów się, w czym AI jest lepsze od Ciebie, a w czym Twoja rola (myślenie strategiczne, kreatywność, relacje międzyludzkie, etyka) pozostaje niezbędna.

Ćwiczenie 3: Przemyśl wpływ AI na Twoją rolę

Zastanów się, które rutynowe, powtarzalne zadania w Twojej przyszłej roli (np. Data Scientist, BI Analyst, Data Engineer) mogą zostać zautomatyzowane lub znacząco wspomagane przez AI. Gdzie Twoje **myślenie strategiczne, kreatywność, umiejętności interpersonalne i odpowiedzialność etyczna** pozostaną kluczowymi umiejętnościami, których AI nie zastąpi?

Quiz na koniec

Quiz “krótkie odpowiedzi”

Odpowiedz na każde pytanie w 2-3 zdaniach. Tak, to takie pytania z rozmowy rekrutacyjnej.

- Jakie są dwa podstawowe narzędzia, które stanowią fundament pracy z danymi?
- Wymień trzy różne typy danych rozpoznawane przez Excel.
- Czym różni się adresowanie względne od bezwzględnego w formułach Excela?
- Jakie jest podstawowe zastosowanie języka SQL w kontekście pracy z danymi?
- Wymień dwa przykładowe typy relacyjnych baz danych SQL.
- Jaki język programowania jest szeroko stosowany w analizie danych i dlaczego jest polecany dla początkujących?
- Podaj dwa przykłady podstawowych koncepcji statystycznych, które są niezbędne w analizie danych.
- Jakie są główne role Data Scientist, Data Engineer i BI Analyst pod względem obszaru zainteresowania?
- Dlaczego budowanie szczegółowego portfolio jest kluczowe dla osób wchodzących do dziedziny analizy danych?
- Wymień dwie kluczowe umiejętności miękkie (interpersonalne) niezbędne w zawodach związanych z danymi.

Klucz odpowiedzi

- Podstawowe narzędzia do pracy z danymi to arkusze kalkulacyjne (jak Excel) i język SQL. Arkusze służą do analizy danych, a SQL do pracy z bazami danych.
- Excel rozpoznaje między innymi typy danych takie jak tekst, liczby oraz daty. Ważne jest zrozumienie, jak Excel interpretuje te typy danych.
- Adresowanie względne (np. A1) zmienia się automatycznie przy kopiowaniu formuł, podczas gdy adresowanie bezwzględne (np. A1) pozostaje niezmienne. Adresowanie mieszane blokuje tylko wiersz lub kolumnę.
- Podstawowe zastosowanie SQL to praca z bazami danych, w tym pobieranie, filtrowanie i manipulowanie dużymi zbiorami danych przechowywanych w relacyjnych bazach.
- Przykładowe typy relacyjnych baz danych SQL to PostgreSQL, MySQL/MariaDB, Microsoft SQL Server i Oracle Database. Każdy z nich ma swoje specyficzne zastosowania.

- Językiem szeroko stosowanym w analizie danych jest Python. Jest przyjazny dla początkujących, a jednocześnie potężny, zwłaszcza dzięki bibliotekom do analizy danych.
- Dwie podstawowe koncepcje statystyczne to miary tendencji centralnej (średnia, mediana, moda) i miary rozproszenia (odchylenie standardowe, wariancja). Pomagają one nadać znaczenie zbiorom danych.
- Data Scientist skupia się na budowaniu modeli predykcyjnych i odkrywaniu wzorców, Data Engineer na budowaniu i utrzymywaniu infrastruktury danych, a BI Analyst na raportowaniu i business insights.
- Budowanie szczegółowego portfolio jest kluczowe, ponieważ pozwala zaprezentować praktyczne umiejętności i pasję, zwłaszcza gdy brakuje formalnego wykształcenia lub doświadczenia zawodowego.
- Dwie kluczowe umiejętności miękkie to rozwiązywanie problemów (identyfikowanie i definiowanie nowych problemów) oraz komunikacja (efektywna wymiana pomysłów i współpraca).

Propozycje pytań "esejowych"

Tutaj spróbuj *popłynąć*. Przygotuj odpowiedzi na 3-5 minut. Spróbuj przygotować je w wersji *dla Anetki z HR*, która nie zna się kompletnie na Twojej pracy oraz dla Twojego przyszłego kierownika - technicznie, z nazwami technologii lub branżowym *slangiem*.

- Opisz rolę i kluczowe obowiązki Inżyniera Danych. Wyjaśnij, w jaki sposób jego praca wspiera analityków danych i data scientists.
- Porównaj plany rozwoju dla Data Scientist i Business Intelligence Analyst, koncentrując się na kluczowych obszarach nauki (np. narzędzia, metody, projekty) i różnicach w profilu roli.
- Wyjaśnij znaczenie statystyki i matematyki w analizie danych. Podaj konkretne przykłady, w jaki sposób podstawowe koncepcje statystyczne (np. miary tendencji centralnej, miary rozproszenia) są wykorzystywane w praktycznej analizie.
- Omów, w jaki sposób praktyczne doświadczenie, takie jak projekty osobiste i staże, przyczynia się do rozwoju kariery w dziedzinie analizy danych. Jakie korzyści płyną z budowania portfolio i uczestnictwa w społecznościach online?
- Szczegółowo opisz jeden z przykładowych projektów portfolio dla Data Scientist (np. predykcja cen nieruchomości, klasyfikacja spamu), przedstawiając jego cel, kluczowe komponenty techniczne, przykładowe źródła danych oraz potencjalne wyzwania.

Zakończenie

Za Tobą **kompletny plan transformacji** – od osoby zaczynającej przygodę z danymi do profesjonalisty gotowego na rynek pracy. Masz już roadmapę, listę umiejętności i wiedzę o tym, czego się nauczyć, gdzie szukać informacji i jak planować rozwój.

Praca z danymi to jeden z najpopularniejszych obecnie kierunków zawodowych – i nie bez powodu. Niezależnie od tego, czy wybierzesz ścieżkę analityka danych, data scientisty, inżyniera danych czy specjalisty BI, czeka Cię satysfakcjonująca kariera pełna różnorodności. Będziesz odkrywać tajemnice e-commerce i marketingu, pomagać bankom walczyć z oszustwami, optymalizować procesy w logistyce, analizować trendy w mediach społecznościowych czy wspierać decyzje w ochronie zdrowia. Każdy projekt to nowa branża, nowe wyzwania, nowe pytania do odpowiedzenia.

Praca z danymi oznacza ciągłe uczenie się – technologie ewoluują, pojawiają się nowe narzędzia, zmieniają się potrzeby biznesu. To, co może wydawać się wyzwaniem, w rzeczywistości jest największą zaletą tego zawodu. Nigdy się nie nudzisz, zawsze masz okazję poznać coś nowego.

Ta książka jako plan powinna Cię wspierać podczas całej podróży. Wracaj do niej, gdy poczujesz się zagubiony, gdy będziesz szukać kolejnego kierunku rozwoju, gdy będziesz potrzebować przypomnienia o fundamentach. Niektóre rozdziały nabiorą sensu dopiero za kilka miesięcy, inne będziesz odkrywać na nowo w miarę zdobywania doświadczenia.

Realistycznie patrząc, **proces transformacji w specjalistę od danych to około roku bardzo intensywnej pracy**. Rok codziennego uczenia się, praktykowania, budowania portfolio, zdobywania pierwszych doświadczeń. To może wydawać się długo, ale pamiętaj – to inwestycja w przyszłość, która zwróci się wielokrotnie.

Nie będzie łatwo. Będą momenty zwątpienia, frustracji, gdy kod nie będzie działać, gdy wyniki analiz będą zaskakujące, gdy nowe narzędzie będzie sprawiać trudności. To normalne. Każdy profesjonalista przeszedł przez te same wyzwania.

Ale gdy już dotrzesz do celu, gdy pierwszy raz Twoja analiza wpłynie na ważną biznesową decyzję, gdy Twój model pomoże rozwiązać realny problem, gdy otrzymasz pierwszą ofertę pracy w wymarzonej roli – wtedy zrozumiesz, że było warto.

Świat danych czeka. Twoja podróż zaczyna się teraz.

Pozostańmy w kontakcie

Jeśli masz pytania, wątpliwości lub chcesz podzielić się swoimi sukcesami na drodze do kariery w analizie danych – nie wahaj się ze mną skontaktować. Każde pytanie jest ważne, każda wątpliwość zasługuje na odpowiedź. Pamiętaj, że wszyscy kiedyś zaczynaliśmy od podstaw.

Znajdziesz mnie w mediach społecznościowych:

- **Facebook**
- **LinkedIn**
- **X (Twitter)**

Dołącz do mojego newslettera - otrzymuj regularne dawki wiedzy o analizie danych, informacje o nowych narzędziach i możliwościach rozwoju bezpośrednio na swoją skrzynkę.

Sprawdź moje inne projekty - znajdziesz tam pogłębione kursy, dodatkowe e-booki i materiały, które pomogą Ci w dalszym rozwoju.

Budowanie kariery w analizie danych to nie solo wyprawa. Społeczność ludzi pracujących z danymi jest otwarta, pomocna i chętna do dzielenia się wiedzą. Wykorzystaj to!

Powodzenia w Twojej podróży z danymi!

Słowniczek

A

- **A/B Testing** - Praktyczne zastosowanie statystyki polegające na testowaniu hipotez i porównywaniu grup (np. różnych wersji kampanii marketingowych, eksperymentów na stronie internetowej, wyników modeli ML) w celu analizy wyników i stwierdzenia, czy różnica jest statystycznie istotna. Kluczowe dla analityków danych. Może też dotyczyć infrastruktury (Data Engineer).
- **Analiza eksploracyjna danych (EDA)** - Kluczowa technika w analizie danych, polegająca na pierwszym spojrzeniu na dane, sprawdzeniu ich jakości, statystyk opisowych, wizualizacji rozkładów i korelacji między zmiennymi. Wykorzystuje statystykę i wizualizacje. Pomaga w odkrywaniu wniosków i insights.
- **Analityk BI (Business Intelligence Analyst)** - Rola w dziedzinie danych, która najłatwiej wejść na rynek. Skupia się na raportowaniu, tworzeniu dashboardów i dostarczaniu business insights. Dużo współpracuje z biznesem. Kluczowe narzędzia to Power BI i Tableau. Wymaga znajomości SQL, narzędzi BI, modelowania danych i zrozumienia biznesu.
- **Apache Airflow** - Biblioteka Pythonowa używana do orchestracji (zarządzania przepływami pracy) potoków danych (DAGs). Kluczowy komponent w implementacji end-to-end ETL pipeline jako narzędzie do schedule'owania.
- **Apache Kafka** - Message Broker używany w systemach strumieniowych w czasie rzeczywistym. Umożliwia zbieranie danych z różnych źródeł (np. eventy ze strony, sensory IoT, logi aplikacji) i publikowanie ich na tematach (Topics).
- **Apache Spark** - Technologia Big Data. PySpark to Python API dla Sparka. Służy do przetwarzania dużych zbiorów danych (transformacje, akcje), posiada interfejs Spark SQL, może być używany do streamingu (Spark Streaming) i wymaga optymalizacji wydajności (partycjonowanie, caching).
- **Arkusze kalkulacyjne** - Podstawowe narzędzia, stanowiące fundament pracy z danymi. Powinny być pierwszym krokiem w nauce. Przykłady to Microsoft Excel, Quip, Zoho Sheet, WPS Spreadsheets. Są niezbędne nawet dla zaawansowanych analityków danych. Umożliwiają nawigację, operacje na danych, formatowanie, organizację danych (sortowanie, filtrowanie, usuwanie duplikatów), użycie podstawowych i zaawansowanych formuł (SUMA, ŚREDNIA, JEŻELI, WYSZUKAJ.PIONOWO, INDEKS+POZYSKAJ), wizualizację danych (wykresy), pracę z tabelami przestawnymi, import i przetwarzanie danych (w tym Power Query) oraz walidację i czyszczenie danych.

B

- **Biblioteki Pythonowe do analizy danych** - Specjalistyczne biblioteki, do których przechodzi się po opanowaniu podstawowej składni Pythona. Kluczowe to pandas (manipulacja, czyszczenie, transformacja danych), NumPy (obliczenia numeryczne, arrays), Matplotlib i Seaborn (wizualizacja danych), scikit-learn (algorytmy uczenia maszynowego).
- **Business Intelligence (BI)** - Dziedzina skupiająca się na raportowaniu i analizie danych biznesowych. Analityk BI intensywnie współpracuje z biznesem. Często wykorzystuje narzędzia takie jak Power BI i Tableau.

C

- **CASE WHEN** - Wyrażenie warunkowe w SQL umożliwiające implementację logiki warunkowej.
- **Certyfikaty** - Formalne potwierdzenie kompetencji w dziedzinie danych (np. Microsoft Certified: Data Analyst Associate, Google Data Analytics Professional Certificate, IBM Data Science Professional Certificate, Tableau Desktop Specialist). Zwiększają atrakcyjność na rynku pracy i pokazują zaangażowanie w rozwój.
- **CI/CD** - Ciągła integracja/ciągłe wdrażanie (Continuous Integration/Continuous Deployment). Praktyki DevOps polegające na automatycznym testowaniu i wdrażaniu, co zapewnia powtarzalność i niezawodność. Przykłady narzędzi to GitHub Actions, GitLab CI. Stosowane np. w dockerized data pipeline.
- **CONSTRAINT** - Ograniczenia w tabelach bazy danych (np. PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL, CHECK, DEFAULT). Zapewniają integralność i poprawność danych.
- **CREATE TABLE** - Instrukcja SQL służąca do tworzenia nowych tabel. Określa nazwy kolumn i ich typy danych.
- **CTE (Common Table Expressions)** - Zaawansowana technika w SQL (definiowana słowem WITH). Umożliwiają tworzenie tymczasowych, nazwanych zbiorów wyników w ramach pojedynczego zapytania. Mogą być rekurencyjne do pracy z danymi hierarchicznymi.

D

- **DAX (Data Analysis Expressions)** - Język formuł używany w narzędziach takich jak Power BI do tworzenia kolumn obliczeniowych i miar.
- **Data Catalog (AWS Glue)** - Centralne miejsce do przechowywania wszystkich schematów danych w Data Lake. Umożliwia automatyczne wykrywanie schematów, śledzi pochodzenie danych, wspiera klasyfikację danych i pomaga w partycjonowaniu.

- **Data Engineer** - Rola w dziedzinie danych skupiająca się na budowaniu i utrzymywaniu infrastruktury danych. Tworzy pipeline'y ETL/ELT, zarządza rozwiązaniami do przechowywania danych (np. data warehouses, systemy real-time), zapewnia dokładność i spójność danych i wykorzystuje technologie big data. Współpracuje z analitykami i Data Scientists.
- **Data Governance** - Procesy i zasady dotyczące zarządzania danymi, obejmujące kontrolę dostępu, bezpieczeństwo, jakość danych, zarządzanie danymi podstawowymi (Master Data Management), dokumentację i wersjonowanie.
- **Data Lake Architecture** - Centralne repozytorium do przechowywania wszystkich danych organizacji w formacie surowym. Umożliwia różne typy analiz. Przykład architektury to AWS Data Lake z S3 + Glue + Athena. Składa się z warstw przechowywania (np. S3), katalogu/schematu (np. Glue) i silnika zapytań (np. Athena).
- **Data Modeling** - Proces tworzenia struktury danych. W kontekście BI często oznacza modelowanie wymiarowe (dimensional modeling), czyli organizację danych w schematy gwiazdy (star schema) lub płatka śniegu (snowflake schema), z tabelami faktów i wymiarów. Kluczowe dla budowania efektywnych raportów.
- **Data Scientist** - Rola w dziedzinie danych, która łączy umiejętności analityczne z zaawansowanym programowaniem, uczeniem maszynowym i modelowaniem. Skupia się na budowaniu modeli predykcyjnych, odkrywaniu wzorców, eksperymentach, wydobywaniu głębszych insights z danych. Wymaga wiedzy z zaawansowanej statystyki, ML, DL, NLP, Computer Vision.
- **DataFrame** - Dwuwymiarowa struktura danych w bibliotece pandas, przypominająca tabelę. Posiada kolumny (Series) i indeks wierszy. Umożliwia łatwą manipulację, czyszczenie, transformację i analizę danych.
- **Debugging** - Proces znajdowania i usuwania błędów w kodzie. W Pythonie można używać prostych instrukcji print() lub debuggera pdb. Kluczowe jest czytanie komunikatów błędów.
- **Deep Learning (DL)** - Zaawansowana dziedzina uczenia maszynowego wykorzystująca sieci neuronowe. Stosowana m.in. w przetwarzaniu obrazów (CNN) i sekwencji danych (RNN/LSTM). Wymaga znajomości narzędzi takich jak TensorFlow/Keras czy PyTorch.
- **Delta Lake** - Format plików kolumnowych dla Data Lake, który zapewnia wersjonowanie danych i wspiera transakcje ACID (atomicity, consistency, isolation, durability).
- **Dimensional Modeling** - Technika modelowania danych często stosowana w hurtowniach danych (Data Warehousing). Polega na organizacji danych wokół tematów biznesowych w postaci tabel faktów (facts) i tabel wymiarów (dimensions), najczęściej w schemacie gwiazdy (star schema).

- **Docker** - Narzędzie do konteneryzacji, które pozwala pakować aplikacje i ich zależności w izolowane środowiska (kontenery). Zapewnia powtarzalność i łatwość wdrożenia. Często używany w połączeniu z Docker Compose do zarządzania wieloma kontenerami. Kluczowe dla DevOps.

E

- **EXPLAIN / EXPLAIN ANALYZE** - Instrukcje SQL służące do analizy planów wykonania zapytania. Pomagają zrozumieć, jak baza danych przetwarza zapytanie i zidentyfikować wąskie gardła w celu optymalizacji wydajności.
- **Excel** - patrz Arkusze kalkulacyjne.

F

- **Feature Engineering** - Proces tworzenia nowych cech (zmiennych) na podstawie istniejących danych w celu poprawy jakości modeli. Może obejmować kalkulacje (np. cena za m^2, wiek budynku), konwersje, transformacje danych. Kluczowy element pracy Data Scientist i w projektach takich jak predykcja cen nieruchomości czy churn klientów.
- **Funkcje agregujące** - Funkcje w SQL (np. COUNT, SUM, AVG, MIN, MAX), które obliczają pojedynczą wartość na grupie wierszy. Wymagają użycia klauzuli GROUP BY. Podobne funkcje dostępne są w Pandas (sum(), mean(), count(), agg()) i NumPy (mean(), median(), std(), var()).
- **Funkcje okna (Window Functions)** - Zaawansowane funkcje w SQL, które wykonują obliczenia na grupie wierszy związanych z bieżącym wierszem (okno), bez zwijania wierszy do pojedynczego wyniku (jak GROUP BY). Przykład to ROW_NUMBER(), RANK(), LAG/LEAD. Używane z klauzulą PARTITION BY do podziału na grupy.

G

- **Git** - System kontroli wersji używany do śledzenia zmian w kodzie i współpracy w zespole. Podstawowe umiejętności obejmują workflow Git.
- **GROUP BY** - Klauzula w SQL używana do grupowania wierszy o tych samych wartościach w określonych kolumnach. Zwykle używana w połączeniu z funkcjami agregującymi.

H

- **HAVING** - Klauzula w SQL używana do filtrowania grup utworzonych przez GROUP BY. Działa podobnie do WHERE, ale na poziomie grup, nie pojedynczych wierszy.

I

- **INDEKS + PODAJ.POZYCJĘ (INDEX + MATCH)** - Funkcje w Excelu umożliwiające bardziej elastyczne wyszukiwanie niż WYSZUKAJ.PIONOWO.
- **INSERT INTO** - Instrukcja SQL służąca do dodawania nowych wierszy (rekordów) do tabeli.

J

- **JOIN** - Klauzula w SQL służąca do łączenia wierszy z dwóch lub więcej tabel. Rodzaje JOIN-ów to m.in. INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL OUTER JOIN, CROSS JOIN. Odpowiada za łączenie danych z różnych źródeł w hurtowni danych. W pandas analogiczne operacje to merge(), join() i concat().
- **Jupyter Notebook / JupyterLab** - Interaktywne środowiska (IDE) idealne do analizy danych i eksploracji. Pozwalają na łączenie kodu Pythona (lub innych języków), wizualizacji i tekstu (Markdown). Kluczowe narzędzie pracy analityka danych i Data Scientist.

K

- **Kafka** - patrz **Apache Kafka**
- **Kaggle** - Platforma online oferująca publiczne zestawy danych, konkursy analizy danych i przestrzenie do ćwiczeń. Jest świetnym miejscem do rozwijania umiejętności, budowania portfolio i rozwiązywania problemów.
- **KPI (Key Performance Indicators)** - Kluczowe wskaźniki wydajności. Metryki używane do mierzenia sukcesu biznesowego lub operacyjnego. Często prezentowane na dashboardach menedżerskich i wykorzystywane w raportach finansowych, sprzedaży, marketingu i operacyjnych.
- **Kubernetes** - system do zarządzania wieloma pojemnikami Docker na raz, automatycznie je uruchamia, restartuje gdy się zepsują, rozdziela ruch między nimi i skaluje w górę lub w dół w zależności od potrzeb.

M

- **Machine Learning (ML)** - Dziedzina sztucznej inteligencji skupiająca się na tworzeniu algorytmów, które uczą się na podstawie danych (np. budowanie modeli predykcyjnych). Kluczowe dla Data Scientist. Wymaga znajomości technik uczenia nadzorowanego i nienadzorowanego, oceny modeli, a także bibliotek takich jak scikit-learn.
- **Matplotlib** - Biblioteka Pythonowa służąca do tworzenia wizualizacji danych. Podstawowy workflow obejmuje figure, axes, plot, show. Umożliwia tworzenie różnych

typów wykresów (liniowy, punktowy, histogram, słupkowy, kołowy, boxplot), formatowanie ich i integrację z pandas.

- **Modelowanie danych (w BI)** - patrz Data Modeling.
- **Modelowanie regresyjne** - Technika statystyczna i uczenia maszynowego służąca do przewidywania wartości ciągłej zmiennej (np. ceny nieruchomości) na podstawie innych zmiennych. Przykładowe modele to regresja liniowa, drzewiaste (XGBoost, LightGBM). Wymaga ewaluacji (np. RMSE, MAE). Podstawy regresji liniowej są częścią statystyki dla analityków.

N

- **NoSQL** - Alternatywne systemy baz danych do relacyjnych. Często stosowane w inżynierii danych dla danych półstrukturalnych lub niestrukturalnych, np. MongoDB, Redis. Różnią się od SQL w składni i modelu danych.
- **NLP (Natural Language Processing)** - Dziedzina zajmująca się przetwarzaniem i analizą języka naturalnego (tekstu). Obejmuje techniki przetwarzania tekstu (czyszczenie, tokenizacja, lematyzacja), ekstrakcję cech (TF-IDF, word embeddings) i budowanie modeli (np. klasyfikacja tekstu, analiza sentymentu). Kluczowa specjalizacja dla Data Scientist.
- **NumPy** - Biblioteka Pythonowa służąca do obliczeń numerycznych. Główną strukturą danych są arrays, które są wydajniejsze od list Pythona. Umożliwia operacje na tablicach (arrays) i zawiera wiele funkcji matematycznych i statystycznych.

O

- **ORDER BY** - Klauzula w SQL służąca do sortowania wyników zapytania. Domyślnie rosnąco (ASC), można też malejąco (DESC).

P

- **Pandas** - Biblioteka Pythonowa niezbędna do manipulacji i analizy danych. Głównymi strukturami danych są Series (jednowymiarowe) i DataFrame (dwuwymiarowe). Umożliwia wczytywanie/zapisywanie danych, eksplorację, indeksowanie/selekcję, czyszczenie/transformację, grupowanie/agregację (groupby(), agg()), łączenie danych (concat(), merge(), join()) oraz tworzenie wizualizacji.
- **Parquet** - Format plików kolumnowych używany w Data Lake. Jest świetny do analityki, ponieważ umożliwia wydajne czytanie tylko potrzebnych kolumn.
- **Partycjonowanie** - Technika optymalizacyjna polegająca na podziale dużych tabel danych (np. w hurtowni danych) na mniejsze, bardziej zarządzalne części (partycje). Stosowane dla wydajności zapytań i zarządzania danymi.

- **Pipeline danych (Data Pipeline)** - Sekwencja kroków (często zautomatyzowanych) do przesyłania, przetwarzania i udostępniania danych z jednego lub wielu źródeł do miejsca docelowego (np. hurtowni danych). Przykład to ETL pipeline (Extract, Transform, Load) lub system real-time streaming. Ich projektowanie i automatyzacja to kluczowy obowiązek Data Engineera. Mogą być skonteneryzowane (Dockerized Data Pipeline).
- **Portfolio projektów** - Zbiór praktycznych projektów stworzonych samodzielnie lub w ramach kursów, prezentujący zdobyte umiejętności. Kluczowe dla osób bez formalnego doświadczenia, aby zaprezentować się potencjalnym pracodawcom. Przykłady projektów podano dla Data Scientist, Data Engineer i BI Analyst.
- **Power BI** - Narzędzie BI od Microsoftu. Składa się z Power BI Desktop (tworzenie raportów), DAX (formuły), Power Query (transformacja danych). Umożliwia modelowanie danych, publikowanie raportów w Power BI Service i łączenie z danymi lokalnymi (Gateway). Kluczowe narzędzie dla BI Analyst.
- **Power Query** - Narzędzie w Excelu i Power BI do automatyzacji czyszczenia i transformacji danych przy użyciu języka M. Używane do wczytywania danych i transformacji.
- **Programowanie w Pythonie** - Kluczowa umiejętność techniczna. Python jest szeroko stosowany w analizie danych, jest przyjazny dla początkujących i jest jednym z najpopularniejszych języków programowania. Podstawy obejmują składnię, typy danych, struktury danych (listy, krotki, słowniki, zbiory), kontrolę przepływu (if, for, while), funkcje. Po podstawach przechodzi się do bibliotek specjalistycznych (pandas, numpy, matplotlib, seaborn, scikit-learn).
- **Projekty osobiste** - Tworzenie własnych projektów analizy danych, często na publicznie dostępnych zbiorach danych, aby zdobyć praktyczne doświadczenie i zbudować portfolio. Stanowią kluczowy element nauki. Przykłady to analiza danych sprzedażowych, pogody, finansów.

R

- **Relacyjne bazy danych** - Typ baz danych, w których dane są zorganizowane w tabele z kolumnami i wierszami, powiązane ze sobą za pomocą kluczy (głównych i obcych). SQL jest językiem do pracy z takimi bazami. Przykłady to PostgreSQL, MySQL, Microsoft SQL Server, Oracle.

S

- **Spark** - patrz **Apache Spark**
- **SQL (Structured Query Language)** - Niezbędne narzędzie, umożliwiające pracę z bazami danych. Umożliwia pobieranie, filtrowanie i manipulowanie danymi przechowywanymi w relacyjnych bazach danych. Podstawy obejmują zapytania

SELECT, FROM, filtrowanie (WHERE, IN, BETWEEN, LIKE, IS NULL), operatory logiczne (AND, OR, NOT), sortowanie (ORDER BY), funkcje wbudowane, funkcje agregujące (COUNT, SUM, AVG), grupowanie (GROUP BY, HAVING), łączenie tabel (JOIN). Zaawansowane tematy to podzapytania, wyrażenia warunkowe (CASE WHEN), funkcje okna, modyfikowanie danych (INSERT, UPDATE, DELETE, TRUNCATE), tworzenie/modyfikacja struktury (CREATE TABLE, ALTER TABLE, DROP TABLE, Constraints, Indexes). Jest standardem w komunikacji z bazami danych i jest niezbędny do integracji z narzędziami BI.

- **Statystyka i Matematyka (Podstawy)** - Solidne podstawy są niezbędne do nadawania znaczenia dużym zbiorom danych. Kluczowe pojęcia to miary tendencji centralnej (średnia, mediana, moda), miary rozproszenia (odchylenie standardowe, wariancja), prawdopodobieństwo i rozkłady, korelacja i regresja, testowanie hipotez. Wiedza ta jest kluczowa do interpretacji wyników, A/B testing, EDA, raportowania i komunikowania niepewności. Poziom opisany w źródłach jest wystarczający do profesjonalnej pracy bez wchodzenia w zaawansowane ML.
- **Staże i Praktyki** - Praktyczne doświadczenie zdobywane w firmie. Jest nieocenione do nauki od doświadczonych profesjonalistów i wglądu w rzeczywiste problemy biznesowe.
- **Subqueries (Podzapytania)** - Zapytania SQL zagnieżdżone wewnątrz innych zapytań. Mogą występować w klauzulach WHERE, SELECT lub FROM.

T

- **Tableau** - Narzędzie BI do wizualizacji i tworzenia dashboardów. Umożliwia tworzenie wykresów, dashboardów, historii, definiowanie pól obliczeniowych, parametrów i filtrów, łączenie z danymi i publikowanie na serwerze/online. Kluczowe narzędzie dla BI Analyst.
- **Tabele przestawne** - Narzędzie w arkuszach kalkulacyjnych (np. Excel), umożliwiające szybką agregację i segmentację danych. Pozwalają na tworzenie podsumowań i wykresów przestawnych.
- **Terraform** - narzędzie do automatycznego tworzenia infrastruktury w chmurze za pomocą kodu
- **Testowanie hipotez** - Technika statystyczna używana do wnioskowania o populacji na podstawie próbki danych. Polega na formułowaniu hipotezy zerowej i alternatywnej i obliczaniu p-value. Podstawy testów (t-Studenta, chi-kwadrat, ANOVA) są ważne dla analityków. Praktyczne zastosowania to m.in. A/B testing i porównywanie grup.

U

- **Uczenie maszynowe (ML)** - patrz Machine Learning.

- **Umiejętności interpersonalne (miękkie)** - Niezbędne obok umiejętności technicznych. Obejmują rozwiązywanie problemów (identyfikacja i definiowanie problemów, radzenie sobie z niepewnością), komunikację (wymiana pomysłów, współpraca, jasne wyrażanie poglądów), umiejętności analityczne (dekonstrukcja informacji, analiza danych, wyciąganie wniosków, krytyczne myślenie) oraz umiejętności badawcze (zdobywanie i rozumienie nowych informacji, ocenianie).
- **UPDATE** - Instrukcja SQL służąca do modyfikowania istniejących wierszy (rekordów) w tabeli. Używa klauzuli WHERE do określenia, które wiersze mają być zmienione.

W

- **Wiedza domenowa** - Zrozumienie dziedziny (branży), w której znajdują się analizowane dane (np. finanse, opieka zdrowotna, e-commerce). Pozwala na wyciąganie bardziej wartościowych wniosków z analiz i może dostarczyć użytecznych spostrzeżeń.
- **WHERE** - Klauzula w SQL używana do filtrowania wierszy na podstawie określonych warunków. Wykorzystuje operatory porównania, operatory logiczne i wzorce tekstowe.
- **Wizualizacja danych** - Proces tworzenia graficznej reprezentacji danych (np. wykresy). Niezbędna do prezentacji wyników analizy, identyfikacji trendów i wzorców oraz storytellingu z danymi. Narzędzia to Excel, Matplotlib, Seaborn w Pythonie, oraz narzędzia BI takie jak Power BI i Tableau.
- **WYSZUKAJ.PIONOWO (VLOOKUP)** - Podstawowa funkcja wyszukiwania w Excelu.

DA DANE i ANALIZY

Lista użytych zdjęć: okładka | strona 2 | strona 9 | strona 22 | strona 46 | strona 75 | strona 76 | strona 77