Veloreciptor

Bogdan Jastrzębski
Mateusz Bąkała

1 czerwca 2020

Spis treści

1 Cel projektu

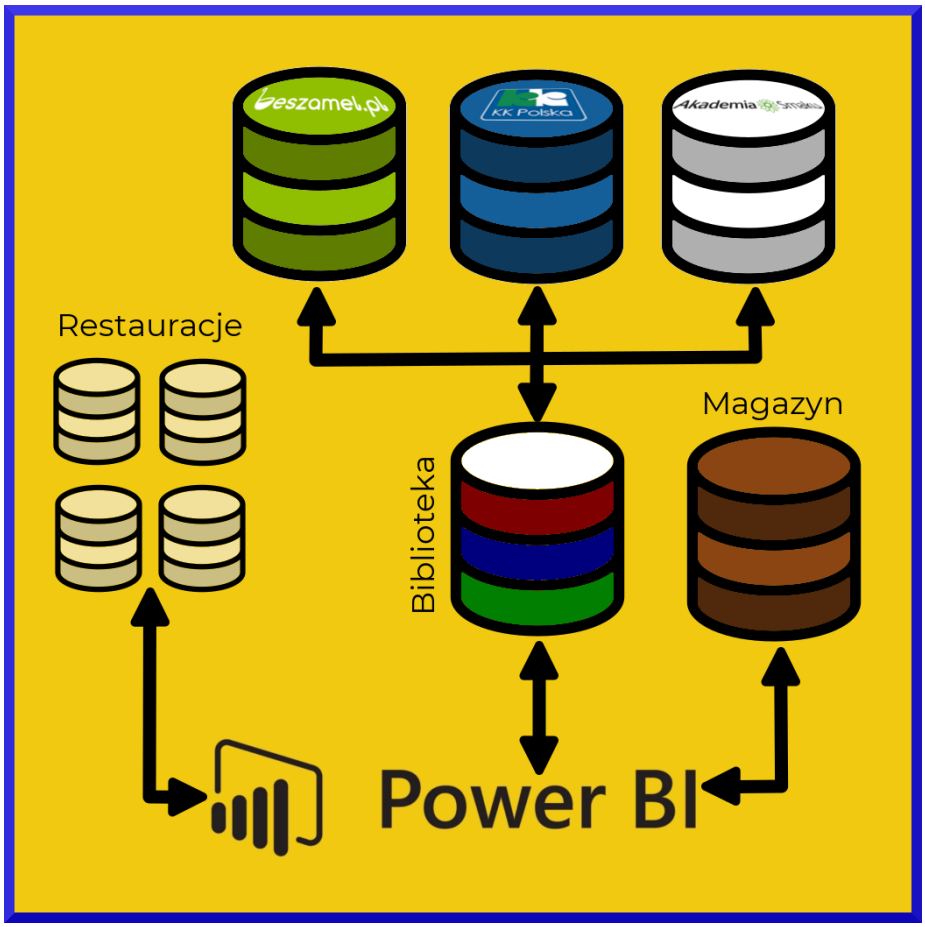
Projekt jest skierowany do właścicieli sieci restauracji, którzy chcieliby wykorzystać generowane dane do optymalizacji strategii oraz analizy zachodzących procesów, w szczególności pod kątem trendów. Naszym celem jest realizacja ich marzeń. Rozwiązanie zakłada istnienie niezerowej liczby restauracji powiązanych z jednym magazynem, przypuszczalnie w obrębie jednego miasta bądź rejonu.

Zdefiniowaliśmy następujące przypadki użycia:

- analiza przyjętych zamówień,
- analiza zawartości magazynu oraz skuteczności wykorzystania produktów w nim przechowywanych,
- analiza rentowności
- identyfikacja trendów okresowych w zachowaniu i nawykach klientów,
- planowanie menu.

2 Architektura rozwiązania

Rozwiązanie składa się z hurtowni danych w technologii Microsoft Power BI przechowującej informacje o przepisach i produktach, zasilanej przez moduł ETL z hipotetycznie dowolnej liczby restauracji oraz jednego magazynu. Ponadto dostępne jest rozwiązanie Business Intelligence w tej samej technologii, zintegrowane z hurtownią.



3 Ekstrakcja danych

Z uwagi na brak dostępu do danych silnie zinformatyzowanych restauracji, pierwszym krokiem było napisanie i uruchomienie symulacji generującej sztuczne zbiory danych. Celem było uzyskanie symulacji eksportu danych z relacyjnej bazy danych.

3.1 Biblioteka przepisów

Dane o przepisach zbierane są z trzech serwisów z przepisami kulinarnymi – KK Polska [4], Beszamel SE [3] oraz Akademia Smaku [1]. Zawierają one pewną liczbę dokumentów HTML dostosowanych do odczytu przez ludzi. Każdy z serwisów ma oddzielną strategię rozmieszczania informacji w obrębie strony, co wymusiło oddzielne traktowanie każdego przypadku. Do ich interpretacji wykorzystaliśmy bibliotekę Scrapy [2] w języku Python.

W przypadku każdego serwisu zaimplementowana została odpowiednia strategia przeszukiwania drzewa stron, pozwalająca możliwie ograniczyć liczbę wyświetlanych do wyłącznie tych, które zawierają przepisy – w większości opierająca się na iterowaniu po podstronach listy wszystkich przepisów. Z uwagi na fakt, że nie dysponujemy gwarancją, że najnowsze przepisy będą dodawane na początku bądź końcu listy, nie implementowaliśmy zatrzymywania przeszukiwania po wystąpieniu pierwszego powtarzającego się przepisu.

Każdy odczytany plik HTML jest interpretowany zgodnie z procedurą opracowaną dla danego serwisu. Rezultatem działania jest plik w formacie .json zawierający oddzielny wpis dla każdego zescrapowanego przepisu. Każdy wpis natomiast zawiera następujący zestaw informacji:

- `link` – adres URL strony (pole wymagane, jednoznacznie identyfikujące przepis),
- `photo_link` – adres URL grafiki reprezentującej przepis (jeśli grafika nie została wykryta, `null`).
- `title` – tytuł artykułu (teoretycznie pole niewymagane, w praktyce na analizowanych serwisach nie występują artykuły bez tytułu)
- `portions` – liczba naturalna wyrażająca liczbę porcji przygotowywaną z podanej ilości składników (w przypadku braku wartości zastępowana zerem)
- `category` – kategoria opisująca generalną przynależność przepisu, np. „sałatka”, „zupa”, „danie wegetariańskie” (w razie braku wartości zastępowana poprzez `null`)
- `ingredients` – lista ciągów tekstowych opisujących odczytane pola składników; z uwagi na niedostatki w schematyczności artykułów, niekiedy zawiera rekordy nie będące opisem składnika; ponadto uzyskane dane są wysoce nieprzetworzone (pole niewymagane, jednak bardzo przydatne do

wartościowej analizy przepisu; w przypadku braku składników opisywane pustą listą)

- `preparation` – wartość tekstowa w postaci ciągu znaków, stanowiąca połączenie akapitów przygotowania przepisu, przy zachowaniu informacji o nowej linii (pole niewymagane, jednak w praktyce zawsze występujące)

Następnym krokiem było przekształcenie uzyskanych plików do biblioteki przepisów, wykorzystywanej przez restauracje podczas ustalania swojej oferty. Każdy wpis został przetworzony w celu normalizacji pola `ingredients` – ponownie w języku Python, przy wykorzystaniu biblioteki Pandas. W rezultacie powstały pliki:

- `recipes.csv` – tabela o polach `id`, `url`, `title` oraz `preparation`, identyfikująca występujące przepisy
- `ingredients.csv` – tabela o polach `id` oraz `name`, identyfikująca występujące składniki
- `recipe_ingredient.csv` – tabela o polach `recipe_id`, `product_id`, `quantity` oraz `measure`, pozwalająca przyporządkować składnik do przepisu oraz określić jego ilość

3.2 Historia zamówień

W celu uzyskania interesujących i miarodajnych wizualizacji, historia zamówień została wygenerowana przy użyciu narzędzi statystycznych (oraz pewnej ilości dżemu truskawkowego). W pierwszym kroku każdemu przepisowi przyporządkowane zostały dwie dodatkowe wartości:

- `frequency` – pochodząca z rozkładu geometrycznego o parametrze $p = 0.4$, oznaczająca popularność przepisu wśród klientów
- `default_price` – pochodząca z rozkładu jednostajnego na przedziale $[44.0, 96.0]$, oznaczająca domyślną, wyjściową cenę przepisu, bez uwzględnienia dodatkowych obniżek bądź podwyżek

Oprócz tego zainicjalizowanie historii zamówień wymagało podania liczby obserwacji `n` oraz dat ograniczających przedział przyjmowania zamówień od dołu i od góry.

Główny proces generacji polegał na wylosowaniu `n` obserwacji ze zwracaniem ze zbioru przepisów, przy uwzględnieniu popularności tychże. Następnie wylosowany został wektor o tej samej długości, ale zawierający daty z podanego przedziału, wybrane z rozkładu jednostajnego. Po uzyskaniu tych informacji, możliwe było stworzenie dodatkowych trendów w zestawie danych:

- Zdefiniowane zostały tzw. „Cheap Fridays", czyli ceny zamówień złożonych w piątki obejmowały rabat 25%.

- Data zamówienia rozszerzona została o godzinę z minutami, losowaną z rozkładu trójkątnego o minimum w godzinie 9:00, maksimum w 21:00 oraz modzie w 15:00.
- Dla weekendów zdefiniowanych jako sobota oraz niedziela moda (czyli godzina szczytu) została przesunięta na godzinę 18:00.

Ostatecznie powstał plik `orders.csv` o 18715 wierszach i następujących kolumnach:

- `id` – numeryczny, jednoznaczny identyfikator rekordu zamówienia
- `recipe_id` – numeryczny identyfikator zamówionego przepisu; dla uproszczenia założyliśmy, że każda zamówiona potrawa stanowi oddzielne zamówienie, co pozostaje zasadniczo bez znaczenia, jako że nie przechowujemy danych na temat klientów, którzy złożyli to zamówienie; nie byłoby to zresztą zbytnio możliwe z uwagi na dyrektywę RODO
- `price` – wartość liczbowa o dokładności do dwóch miejsc po przecinku, stanowiąca kwotę zapłaconą za zamówienie
- `date` – tekstowa reprezentacja daty i czasu identyfikowanego z zamówieniem (jego złożeniem bądź realizacją, w zależności od przyjętego standardu)

3.3 Historia magazynu

Kluczową kwestią dla pewnej części funkcjonalności było zbudowanie modelu magazynu. Założyliśmy, że dane na temat magazynu mają postać tabeli, w której każdy rekord opisany jest przez następujące pola:

- `id` – numeryczny, jednoznaczny identyfikator rekordu jednostki magazynowej
- `entity_id` – analogicznie do poprzedniego, z tą jednak różnicą, że to pole odnosi się do całej encji jednostki magazynowej, niezależnie od późniejszych jej podziałów
- `product_id` – numeryczny identyfikator produktu stanowiącego daną jednostkę magazynową
- `quantity` – ilość produktu w danej jednostce magazynowej, wyrażona w pewnej z góry założonej jednostce miary, nieistotnej z punktu widzenia większości użytkowników (pod warunkiem, że spójnej); w przypadku ułamków wyrażona w postaci `a/b`, gdzie `a` i `b` to pewne liczby naturalne
- `price` – wartość jednej jednostki produktu w momencie i warunkach zakupu danej jednostki magazynowej

- `from_date` – moment umieszczenia danej jednostki magazynowej w magazynie wyrażony poprzez datę bez informacji o godzinie
- `to_date` – moment usunięcia danej jednostki magazynowej z magazynu wyrażony poprzez datę bez informacji o godzinie; jeśli nadal tam jest, pusta wartość
- `exp_date_days` – liczba dni, po których dana jednostka magazynowa musi zostać wyrzucona z powodu przeterminowania się; jako początkowy punkt odniesienia traktuje się pole `from_date`

Aby historia magazynu miała powiązanie z historią zamówień, proces generowania magazynu podzieliliśmy na dwa kroki: „wypełnienie" oraz „przepełnienie". Pierwszy z nich polegał na odczytaniu ilości produktów potrzebnych do zrealizowania zamówień w danym dniu. Na tej podstawie czas `to_date` ustawiany był na zgodny z polem `date` zamówienia, zaś czas `from_date` był ustawiany indywidualnie dla każdego produktu, losując liczbę dni do odjęcia z rozkładu jednostajnego na przedziale $[1, \texttt{exp\_date\_days}]$, przy czym `exp_date_days` zastępowany był przez 365, jeśli przekraczał tę wartość (albowiem trudno oczekiwać, że np. ktokolwiek kupiłby paczkę soli na dwadzieścia lat przed otworzeniem restauracji).

Uzyskana w ten sposób historia magazynu nie zawierała była jednak produktów odrzuconych z powodu przeterminowania ani produktów wciąż obecnych w magazynie. W tym celu przeprowadzone zostało tzw. „przepełnienie". W tym kroku wygenerowane zostało 20000 rekordów z datami pobranymi losowo z tych uzyskanych w pierwszym kroku (przy czym z prawdopodobieństwem proporcjonalnym do częstości występowania tej daty), zaś produktami i cenami wybranymi losowo z listy dostępnych produktów. Czas `to_date` został ustawiony na jeden dzień do przeterminowaniu się produktu, chyba że wykraczałby on poza zakres zbioru danych, wówczas zastępowany był przez `None`, brak wartości. Oprócz tego ilość produktu została rozlosowana z rozkładu geometrycznego z parametrem $p = 0.025$.

Finalnie plik został wyeksportowany jako `warehouse.csv`.

4 Transformacja i integracja danych

Niniejsza część dokumentacji poświęcona jest opisowi procesu transformacji danych, trzeciemu w kolejności w ETL (Extract, Load, Transform). Proces przetworzenia następujących tabel zostanie opisany:

- `orders.csv`
- `magazine.csv`
- `products.csv`
- `recipes.csv`
- `recipe_ingredients.csv`

4.1 Załadowanie i transformacja zbioru „orders”

4.1.1 Dane wejściowe

Dane wejściowe składają się z następujących kolumn:

- `id`: Liczba całkowita, przedstawia unikalne id rekordu w tabeli.
- `recipe_id`: Liczba całkowita, przedstawia zamówione danie.
- `price`: Liczba zmiennoprzecinkowa, przedstawia cenę, za którą sprzedano danie.
- `date`: Data z godziną, przedstawia datę i godzinę sprzedaży

4.1.2 Kroki transformacji

Transformacja zbioru zawiera następujące kroki:

- zaciągnięcie danych źródłowych,
- zmiana nagłówków,
- zmiana zmiana typów kolumn,
- wstawienie daty wyekstrahowanej z kolumny `date`,
- wstawienie godziny wyekstrahowanej z kolumny `date`,
- usunięcie kolumny `date` i kolumny `id`

Dane czasowe były w zapisane z dokładnością do godziny. Ponieważ w zbiorze posługujemy się głównie datą, bardzo rzadko godziną, kolumna czasu została podzielona na datę i godzinę w postaci dwóch kolumn.

4.1.3 Dane wyjściowe

Ostatecznie dane mają następujące kolumny:

- `recipe_id`: bez zmian,
- `price`: bez zmian,
- `Data`: wyekstrahowana data,
- `Godzina`: wyekstrahowana godzina.

4.1.4 Dodane kolumny i miary

Dodane kolumny to:

- `koszt zamówienia`: przedstawia sumaryczny średnią sumę cen składników. Wyliczany jest za pomocą iloczynu `quantity` z tabeli `recipe_ingredient` i `cena produktu do przepisu`.

4.2 Załadowanie i transformacja zbioru „magazine”

Dane wejściowe składają się z następujących kolumn:

- `product_id`: liczba całkowita, przedstawia ID produktu,
- `quantity`: liczba całkowita, ilość towaru,
- `price`: liczba zmiennoprzecinkowa, cena towaru,
- `exp_date_days`: czas trwania, czas do utraty daty ważności,
- `from_date`: data, data wstawienia do magazynu,
- `to_date`: data, data wyjęcia z magazynu,
- `id`: liczba całkowita, unikalne id,
- `entity_id`: liczba całkowita, unikalne ID partii towaru zakupionego jednorazowo.

4.2.1 Kroki transformacji

Transformacja zbioru zawiera następujące kroki:

- zaciągnięcie danych źródłowych,
- zmiana nagłówków,
- zmiana zmiana typów kolumn,
- zmiana typu `exp_date_days`,

- dodanie kolumny `expiry_date`,
- zmiana typu kolumny `expiry_date`,
- dodanie kolumny `expired`,
- zmiana typu kolumny `expired`,
- zmiana nazwy kolumny `expired`,
- podział kolumny `quantity` według znaku `/`,
- zmiana `null` na 1 w kolumnie `quantity.2`,
- zmiana wartości „“ na 0 w kolumnie `quantity.1`,
- zmiana typu `quantity.1` i `quantity.2`,
- usunięcie kolumn `exp_date_days` i `expiry_date`,
- anulowanie przestawienia kolumn,
- zmiana nazwy `atrybut` na `action`,
- zmiana nazwy `wartość` na `date`,
- zmiana wartości `from_date` na `in`,
- zmiana wartości `to_date` na `out`,
- wstawienie kolumny powstałej przez podzielenie `quantity.1` przez `quantity.2`,
- wstawienie dzielenia,
- usunięcie kolumn,
- zmiana nazwy

Wejściowy format danych przewidywał jeden rekord zarówno na wpisanie i wypisanie produktu. W tej formie dane były trudne do przetwarzania. Zostały przetransformowane w dane, które wyrażają zmiany w magazynie. Dzięki temu, bardzo łatwo można wygenerować później dane dzienne poprzez kumulowanie danych. Większość wierszy została podzielona na dwa. Jeden wyraża wprowadzenie danej partii produktu do bazy danych, drugi na wyjęcie danej partii produktu.

4.2.2 Dane wyjściowe

Ostatecznie dane mają następujące kolumny:

- `product_id`: liczba całkowita, przedstawia ID produktu,
- `price`: liczba zmiennoprzecinkowa, cena towaru,
- `quantity`: liczba całkowita, ilość towaru,
- `expiry_date`: data ważności,
- `from_date`: data, data wstawienia do magazynu,
- `to_date`: data, data wyjęcia z magazynu,
- `id`: liczba całkowita, unikalne id,
- `entity_id`: liczba całkowita, unikalne ID partii towaru zakupionego jednorazowo.

4.3 Załadowanie i transformacja zbioru products

Dane wejściowe składają się z następujących kolumn:

- `name`: napis zmiennej długości, nazwa produktu,
- `vegetarian`: czy jest wegetariański,
- `vegan`: czy jest wegański,
- `allergic`: czy jest alergenna,
- `gluten`: czy zawiera gluten,
- `exp_date_days`: przeciętny czas upływu ważności,
- `price`: cena,
- `id`: ID produktu,
- `Klasa`: klasa ogólna produktu,
- `Kategoria`: kategoria, podklasa produktu.

4.3.1 Kroki transformacji

Transformacja zbioru zawiera następujące kroki:

- zaciągnięcie danych źródłowych,
- zmiana nagłówków,
- zmiana zmiana typów kolumn,
- zmiana typu `exp_date_days` na `czas trwania` i `price` na liczbę zmiennoprzecinkową,
- zmiana nazwy kolumny `id` na `product_id`,
- zmiana wartości „“ w kolumnie `Kategoria` na `null`.

Tabela produktów była od początku prawie gotowa do załadowania. Zostały naniesione tylko drobne poprawki w typie danych i nazwach, by były spójne w bazie danych.

4.3.2 Dane wyjściowe

Ostatecznie dane mają następujące kolumny:

- `name`: bez zmian,
- `vegetarian`: bez zmian,
- `vegan`: bez zmian,
- `allergic`: bez zmian,
- `gluten`: bez zmian,
- `exp_date_days`: bez zmian,
- `price`: bez zmian,
- `product_id`: wcześniej `id`, bez zmian,
- `Klasa`: bez zmian,
- `Kategoria`: bez zmian.

4.3.3 Dodane kolumny i miary

Dodana została nowa hierarchia:

- `Klasa`
- `Kategoria`
- `Produkt`

4.4 Załadowanie i transformacja zbioru recipes

Dane wejściowe składają się z następujących kolumn:

- `id`: liczba całkowita, ID przepisu,
- `url`: napis zmiennej długości, url przepisu,
- `title`: napis zmiennej długości, nazwa przepisu, `preparation`: napis zmiennej długości, opis przygotowania.

4.4.1 Kroki transformacji

Transformacja zbioru zawiera następujące kroki:

- zaciągnięcie danych źródłowych,
- zmiana nagłówków,
- zmiana zmiana typów kolumn,
- usunięcie kolumn `url` i `preparation`,
- zmiana nazw kolumn.

Tabela przepisów zawierała w połowie kolumny niepotrzebne, które zostały usunięte. Jedyną użyteczną kolumną okazała się ostatecznie nazwa produktu i odpowiadające jej ID.

4.4.2 Dane wyjściowe

Ostatecznie dane mają następujące kolumny:

- `recipe_id`: wcześniej `id`, bez zmian,
- `title`: bez zmian,

4.5 Załadowanie i transformacja zbioru recipe_ingredients

Dane wejściowe składają się z następujących kolumn:

- `recipe_id`: ID przepisu,
- `product_id`: ID produktu,
- `quantity`: ilość produktu dla danego przepisu,
- `measure`: jednostka.

4.5.1 Kroki transformacji

Transformacja zbioru zawiera następujące kroki:

- zaciągnięcie danych źródłowych,
- zmiana nagłówków,
- usunięcie kolumny `measure`,
- zmieniono wartość `,` na `.` w kolumnie `quantity`,
- zmiana zmiana typów kolumn,

Tabela `recipe_ingredients` wyraża powiązanie między tabelą produktów i tabelą przepisów. Można postrzegać ją jako tabelę faktów.

4.5.2 Dane wyjściowe

Ostatecznie dane mają następujące kolumny:

- `recipe_id`: bez zmian,
- `product_id`: bez zmian,
- `quantity`: odpowiednio przekonwertowana.

4.5.3 Dodane kolumny i miary

Dodana została kolumna cena produktów. Jest to średnia cena produktu dla danego przepisu. Ta kolumna służy do wyliczania średnich kosztów związanych ze składnikami na przepis.

4.6 Tabela `Calendar`

Tabela `Calendar` została wygenerowana przez program. Ze względu na to, że przede wszystkim w tym projekcie analizujemy przebiegi czasowe, jest to de facto tabela najważniejsza. Zostały zdefiniowane dla niej miary:

- Ilość produktów w magazynie
- Koszty zużytych produktów
- Minus koszty zużytych produktów
- Liczba zamówień
- Wartość magazynu
- Wartość wyrzuconych produktów
- Wejścia na magazyn

- Wydatki z magazynu
- Zysk dzienny miara
- Zysk skumulowany

i kolumny:

- Zysk dzienny
- nr. dnia tygodnia
- Dzień tygodnia

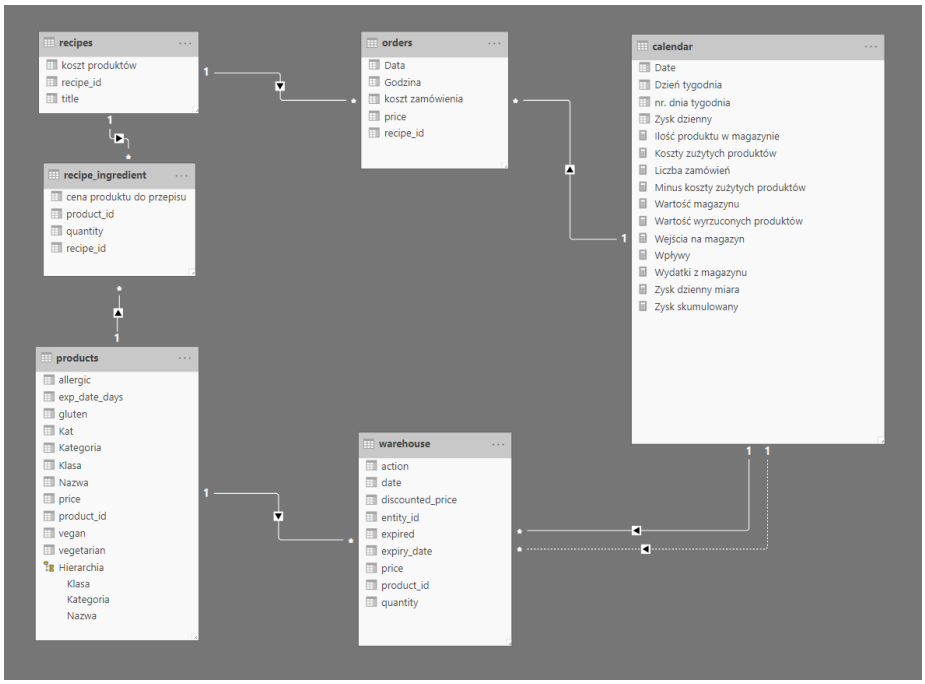
4.7 Ostateczne tabele i ich integracja

Lista wszystkich tabel w programie przedstawia się następująco:

- `orders`
- `magazine`
- `products`
- `recipes`
- `recipe_ingredients`
- `calendar` - tabela wygenerowana przez program

5 Model hurtowni danych

Jak to powiedział kiedyś pewien mądry człowiek, jeden obraz jest wart więcej niż tysiąc słów, pod warunkiem, że jego rozmiar przekracza 10 * n kB, gdzie n oznacza liczbę bajtów używanych do opisu jednego znaku w tych słowach. Dlatego model hurtowni prezentujemy na schemacie graficznym (szczególnie dlatego, że tabele opisane zostały szczegółowo już w poprzednim rozdziale). Tabele łączone były według kolumn `X_id` o wspólnych nazwach (i kolumnie `date` w przypadku tabeli `calendar`).



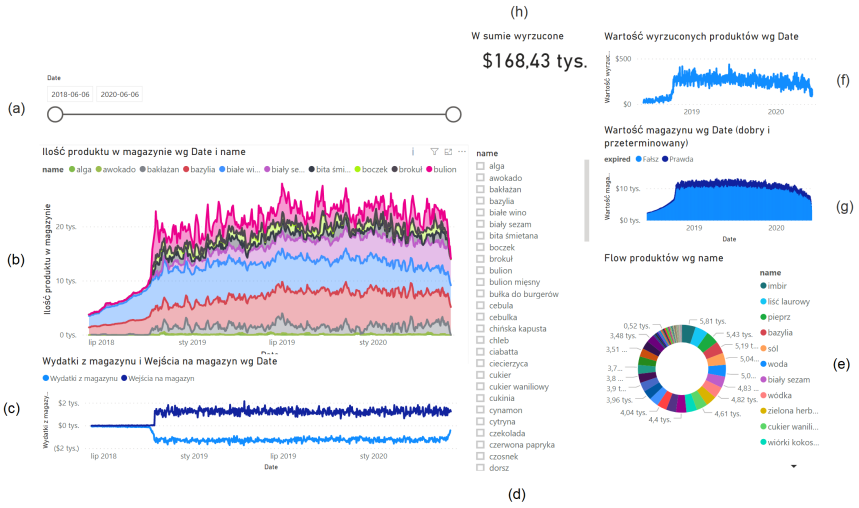
Tabelami faktowymi w tym modelu są z pewnością tabele `orders` oraz `warehouse`, teoretycznie taką tabelą mogłoby być też `recipe_ingredient`, ale nie istnieje wizualizacja wykorzystująca tę tabelę jako główną tabelę faktową.

Najrozsądniejsze wydaje się odświeżanie codziennie, jako że wiele danych i wizualizacji skupia się na zachodzeniu zdarzeń w ciągu jakiegoś dnia, bez równoczesnego uwzględniania godziny. Ponadto większa częstotliwość odświeżania danych nie poprawiłaby możliwości obserwowania trendów. Z uwagi na konstrukcję tabel, w większości tabel możliwe jest ich odświeżanie poprzez przyłączenie nowych rekordów bez zmiany poprzednich. Jedynym wyjątkiem jest tabela `warehouse`, w której konieczne jest zastąpienie wpisów o polu `action = out` i pustej wartości `date` poprzez odpowiadające im co do `entity_id` nowe rekordy, a dokładniej wyciągnięte z nich pola `to_date` i `quantity`.

6 Opis warstwy raportowej

Z uwagi na integrację hurtowni danych oraz warstwy raportowej, dostęp do danych z hurtowni został skonfigurowany do przebiegania w sposób automatyczny. Sam raport podzielony został na pięć tematycznie pogrupowanych paneli:

6.1 Analiza magazynu



Pierwszy panel odnosi się do przypadku użycia, w którym celem jest obserwacja stanu magazynu na przestrzeni czasu.

Suwak (a) pozwala na wybór zakresu dat, który prezentowany jest wykresach tego panelu.

Wykres liniowo-powierzchniowy (b) pokazuje ilość produktów dostępnych w magazynie na przestrzeni dni z podziałem na produkty. Dzięki zastosowaniu tego typu wykresu można jednocześnie stwierdzić też, ile było sumarycznie produktów w magazynie dowolnie wybranego dnia, a także zaobserwować, czy z powodu wzrostu ilości produktów potrzebna będzie inwestycja w przestrzeń magazynową.

Wykres liniowy (c) opisuje koszt produktów włożonych w magazyn oraz „zysk" rozumiany jako odzyskanie wpłaconego kosztu w momencie wyjęcia produktu z magazynu. Pozwala to dostrzegać nieprawidłowości pochodzące z prób defraudacji.

Lista wyboru (d) pozwala wybrać podzbiór produktów, które analizowane są na wykresach tego panelu.

Wykres pierścieniowy (e) pozwala zaobserwować, które produkty najczęściej są składowane w magazynie, dzięki czemu można dostosować przestrzeń magazynową do ich potrzeb.

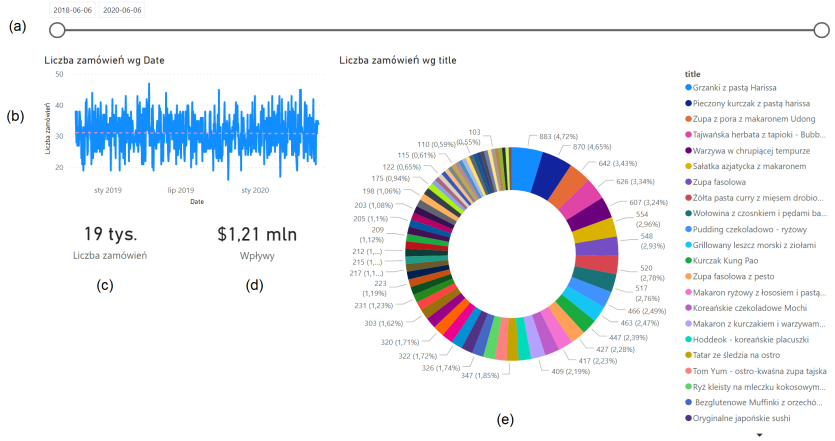
Wykres liniowy (f) demonstruje koszty ponoszone z racji marnotrawienia

zmagazynowanych produktów, co pozwala również wykryć próby defraudacji bądź podkradania produktów, jak również ocenić efektywność pracowników odpowiedzialnych za kontrolę zbliżania się terminu spożycia.

Wykres liniowo-powierzchniowy (g) obrazuje wartość produktów przechowywanych w magazynie na przestrzeni czasu, z podziałem na produkty nadające się do spożycia oraz przeterminowane. Ponownie umożliwia to wykrycie nieprawidłowości, jak również pomiar efektywności.

Ostatnia jest wartość liczbowa (h), stanowiąca swoisty odpowiednik licznika długu publicznego, mającego przypominać o upływie czasu i nietrwałości dóbr doczesnych.

6.2 Analiza zamówień



Drugi panel odnosi się do celu biznesowego, którego sensem jest obserwacja statystyk dotyczących sumarycznie ujętych zamówień.

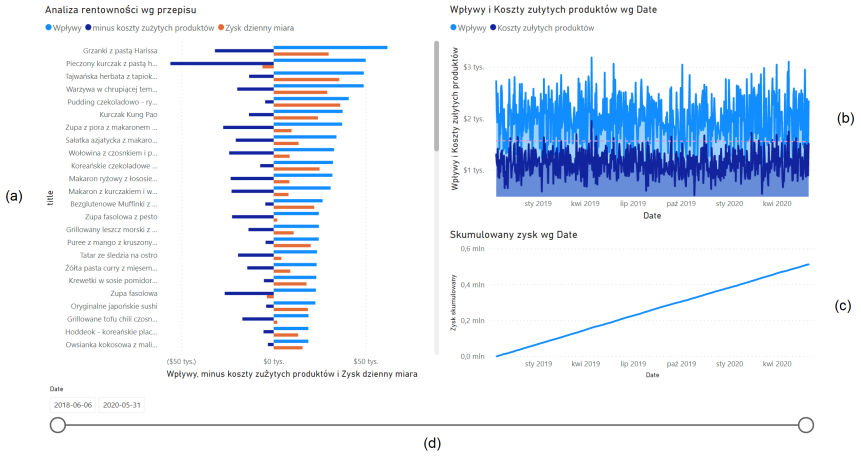
Suwak (a) pozwala na wybór zakresu dat, który prezentowany jest wykresach tego panelu.

Wykres liniowy (b) pozwala obserwować dzienne zmiany liczby zamówień złożonych przez klientów, dopasowuje też linię trendu, co ułatwia zaobserwowanie problemów związanych z popularnością lokalu.

Wartość liczbowa (c) podsumowuje pełną liczbę zamówień, podczas gdy wartość (d) opisuje wartość tych zamówień, bezpośrednio przekładającą się na przychody od klientów.

Wykres pierścieniowy (e) obrazuje procentowy udział danej potrawy wśród wszystkich zamówień, co pozwala ograniczyć koszty związane z utrzymywaniem obszernego menu poprzez redukcję mało popularnych dań.

6.3 Analiza rentowności



Trzeci panel opisuje kluczową z punktu widzenia zarządu i wszelkich osób, które będą miały dostęp do tej analizy w trakcie normalnego toku użytkowania, czyli ogólnie pojętą rentowność.

Wykres słupkowy (a) jest potężnym narzędziem do identyfikacji zbyt tanich dań oraz zwyczajnie nierentownych przepisów, które należałoby usunąć z menu.

Wykres liniowo-powierzchniowy (b) demonstruje wzajemną relację przychodów ze sprzedanych dań oraz kosztów ponoszonych na produktach. Pozwala zidentyfikować moment, w którym konieczne jest podniesienie cen z powodu wyższych cen składników.

Wykres liniowy (c) powinien być wykresem funkcji niemalejącej, jako że przedstawia pieniądze, które wygenerowała restauracja w danym przedziale swojej działalności.

Suwak (d) pozwala na wybór zakresu dat, który prezentowany jest wykresach tego panelu. Ponownie.

6.4 Analiza rytmu

Czwarty panel wspomaga obserwację rytmu dnia i tygodnia, w którym porusza się restauracja za pośrednictwem klientów.

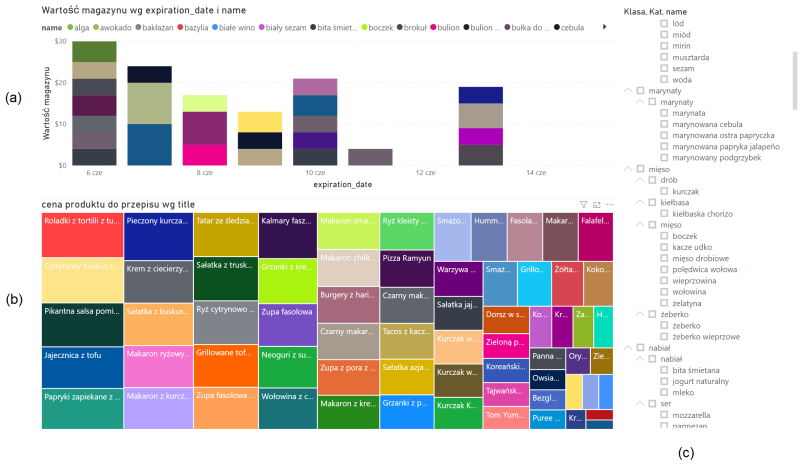
Wykres słupkowy skumulowany (a) pokazuje popularność przepisu według godzin, jak również ogólną liczbę dań sprzedanych o tej godzinie. Umożliwia to przygotowywanie odpowiednich dań z wyprzedzeniem, antycypując nadciągające niczym Zeus zamówienie.

Wykres słupkowy skumulowany (b) demonstruje rzecz analogiczną, lecz w przekroju dni tygodnia, i również może zostać wykorzystany do dalszej poprawy modelu predykcyjnego (chociaż zbudowanie drzewa decyzyjnego na podstawie danych zamiast wykresów może być skuteczniejsze).

Zestawienie słupkowo-tabelaryczne (c) pozwala wygodnie odczytać dokładne liczby zamówień posortowane według popularności dań. Jest to przede wszystkim statystyka dobrze dopełniająca pozostałe wykresy na tej stronie.

Suwak (d) pozwala na wybór zakresu dat, który prezentowany jest wykresach tego panelu. I jest jak stary pies, który zna jedną sztuczkę, ale sztuczkę przydatną w każdej sytuacji życiowej, w której kiedykolwiek się znajdziesz.

6.5 Ustalanie menu



Ostatni, piąty panel służy ustalaniu codziennego menu, w szczególności tzw. dania dnia, poprzez obserwację stanu magazynu.

Wykres słupkowy skumulowany (a) opisuje wartość produktów, które zepsują się danego dnia – z podziałem na produkty. Jest to również pośredni panel sterowania dla wykresu (b).

Lista (c) wyświetla zhierarchizowane produkty występujące w przepisach. Podział na kategorie powinien znacząco przyspieszać przeszukiwanie drzewa. Jest to również bezpośredni panel sterowania.

Mapa drzewiasta (b) prezentuje przepisy z powierzchnią odpowiadającą wartości składników spośród tych, które znajdują się wśród wkrótce przeterminowujących się produktów, co pozwala ocenić wartość przepisu w kontekście wykorzystania psujących się składników (ale jeszcze nie zepsutych). W przypadku zaznaczenia konkretnego składnika bądź kilku składników na którymkolwiek panelu sterującym lista uaktualnia się, wyświetlając przepisy zawierające przynajmniej jeden z zaznaczonego podzbioru składników, ponownie przeliczając wartość przepisu w nowym kontekście.

7 Podsumowanie rezultatów

Trudno jednoznacznie ocenić możliwości zastosowania rozwiązania w praktyce. Głównym problemem jest brak silnie zautomatyzowanych sieci restauracji, ponieważ zebranie wszystkich potrzebnych danych wymagałoby znaczącego zwiększenia nakładów pracy wymaganych od pracowników w podstawowych czynnościach albo inwestycji w wysoce zaawansowane rozwiązania z pogranicza uczenia maszynowego. Ponadto, gdyby już taka sieć istniała, z dużym prawdopodobieństwem przechowywać będzie dane w inny sposób niż ten przez nas założony.

Gdyby jednak spełnione zostały wszystkie wymagane warunki działania, uzyskiwane raporty wydają się być użyteczne w zarządzaniu siecią restauracji, w szczególności ostatnia karta raportu jest nowatorską ideą, wykorzystującą możliwości dawane przez specyficzny zestaw danych. Oczywiście, wszystko działałoby do momentu, w którym kucharz zmarnowałby jakiś składnik podczas gotowania, klient wyszedł bez płacenia albo poprosił o danie bez któregoś z alergenów.

8 Podział pracy

Na początku należy zaznaczyć, że obaj autorzy mieli znaczący wkład w przygotowanie omawianego rozwiązania i żaden nie rości sobie praw do określenia tego projektu jego samodzielnym dziełem.

Pan Jastrzębski odpowiedzialny był za implementację projektu na platformie Microsoft Power BI, w szczególności transformację i ładowanie danych, a także design raportu, jak również etykietowanie danych użytych do przetestowania działania projektu.

Pan Bąkała odpowiedzialny był za przygotowanie skryptów pobierających i generujących dane, jak również zdalne konsultacje w fazie implementacji projektu w ramach tzw. „extreme programming" i drugą część etykietowania danych.

Temat projektu, dokumentacja oraz debugging są wspólnym dziełem i bezcelowe jest wskazywanie odpowiedzialności za każdą część składową tychże.

Bibliografia

[1] *Akademia Smaku*. URL: https://akademiasmaku.pl/. (dostęp: 22.05.2020).

[2] *Scrapy | A Fast and Powerful Scraping and Web Crawling Framework*. URL: https://scrapy.org/. (dostęp: 22.05.2020).

[3] SUPERexpress. *Przepisy kulinarne i gotowanie - Beszamel.se.pl*. URL: https://beszamel.se.pl/. (dostęp: 22.05.2020).

[4] KK Polska Sp. z.o.o. *KK Polska*. URL: https://kkpolska.pl/. (dostęp: 22.05.2020).