Który silnik bazy do czego: przewodnik decyzyjny

Poziom: podstawy **Dotyczy: wszystkie silniki** **Czas: ~15 min**

Zielona ma naraz osiem różnych silników baz danych, a żaden z nich nie stoi tam dla samej różnorodności. Ta strona pokazuje, jak myśleć o wyborze silnika, zanim padnie pytanie, dlaczego akurat ten, a nie inny.

Skróty użyte na tej stronie (6)

Skrót	Co znaczy
CSV	w Windowsie: Cluster Shared Volume, dysk widoczny naraz dla wszystkich węzłów klastra; w danych: Comma-Separated Values, plik z wartościami oddzielonymi przecinkami
HANA	SAP HANA: baza danych w pamięci firmy SAP, używana do dużych systemów księgowych i magazynowych
IT	Information Technology: dział albo praca związana z komputerami i infrastrukturą firmy
MSSQL	Microsoft SQL Server: silnik bazy danych Microsoftu, na nim w Zielonej stoi księgowość Optima
SAP	Systeme, Anwendungen und Produkte in der Datenverarbeitung: niemiecki producent oprogramowania klasy ERP, właściciel silnika HANA i przejętego wcześniej Sybase ASE
WMS	Warehouse Management System: system zarządzania magazynem

Z życia wzięte

Z ŻYCIA WZIĘTE

Pan Zbigniew, po powrocie z konferencji, pyta na spotkaniu: „słyszałem, że jedna nowoczesna baza załatwia wszystko, po co nam osiem różnych?". Odpowiedź nie mieści się w jednym zdaniu: Optima przyszła z wymogiem MSSQL, WMS z wymogiem Oracle, a żadna z tych decyzji nie zapadła w dziale IT.

Po co to istnieje

Pytanie „jaki silnik bazy wybrać" pada rzadziej, niż mogłoby się wydawać, bo w większości przypadków wyboru w ogóle nie robi administrator. Aplikacja pudełkowa przychodzi z konkretnym silnikiem wpisanym w umowę licencyjną, a decyzja o silniku zapadła u producenta tej aplikacji, czasem lata przed tym, jak trafiła do naszej firmy. Realny wybór administratora dotyczy głównie własnych, wewnętrznych narzędzi: wtedy warto mieć uporządkowany sposób myślenia, zamiast sięgać po silnik, który akurat ostatnio ktoś chwalił.

Skąd bierze się różnorodność

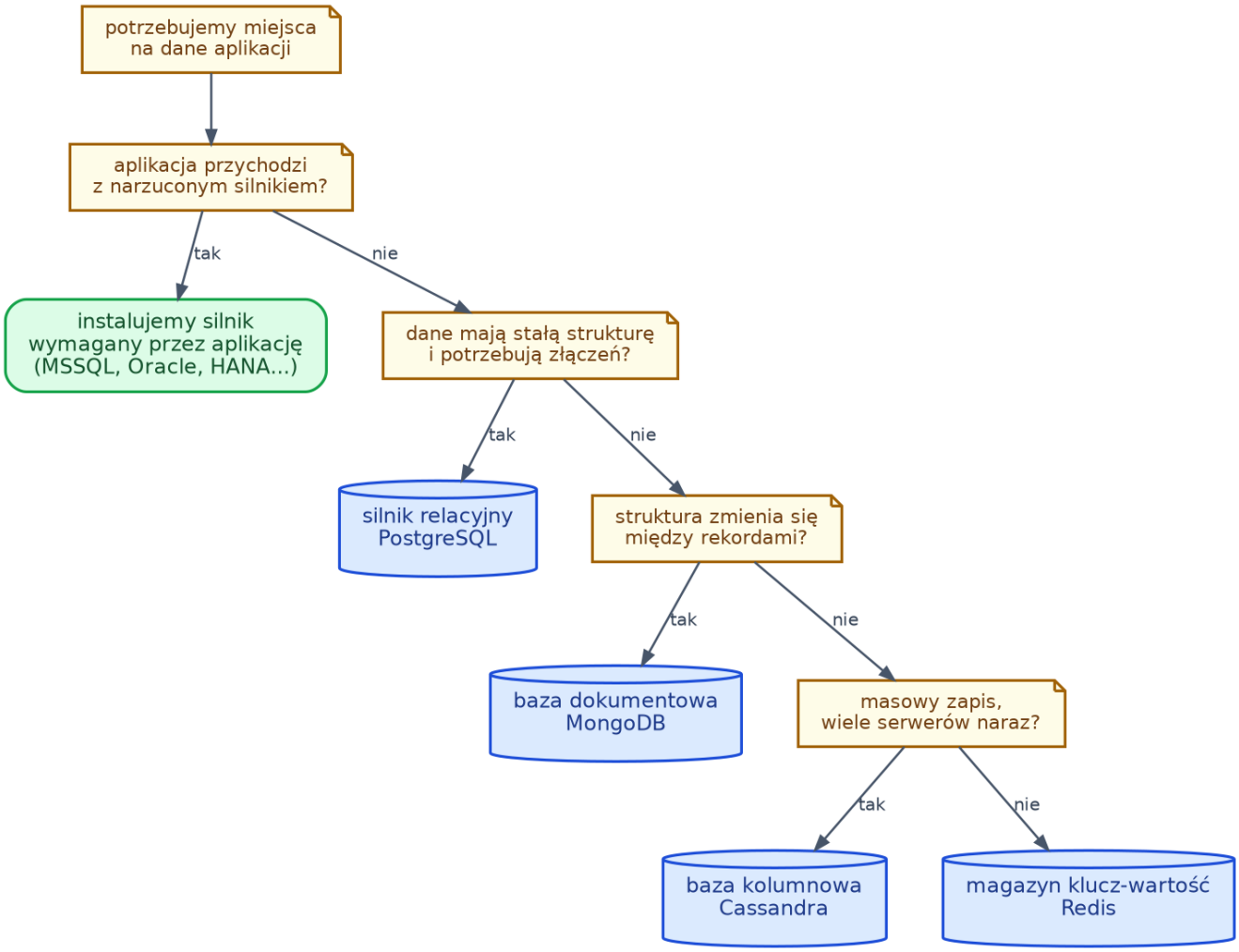
Potrzeba	Silnik	Dlaczego	Przykład w Zielonej
Gotowa aplikacja pudełkowa (księgowość, kadry) wymaga konkretnego silnika w licencji	Ten, który narzuca producent (tu: MSSQL)	Wyboru dokonał dostawca aplikacji, nie dział IT; zmiana silnika oznaczałaby zmianę całej aplikacji	Optima na `sql01`, baza `CDN_Zielona`
System od zewnętrznego dostawcy z własnym wymogiem silnika	Silnik wskazany w umowie wdrożeniowej (tu: Oracle)	Jak wyżej: silnik jest częścią produktu, nie osobną decyzją	WMS na `ora01`, baza `WMSPRD`
Własna aplikacja pisana wewnątrz firmy, dane o stałej strukturze i częstych złączeniach	Silnik relacyjny bez przymusu licencyjnego (tu: PostgreSQL)	Darmowy, dobrze udokumentowany, zespół programistów już go zna z poprzednich projektów	Aplikacje zespołu Oli i baza Zabbixa na `pg01`
Gotowy produkt (sklep, CRM) kupiony razem z konkretnym silnikiem w komplecie	Silnik, z którym produkt przyszedł (tu: MariaDB)	Podobnie jak w księgowości, decyzja zapadła w chwili zakupu gotowego produktu	Sklep B2B na `my01`, baza `sklep_b2b`
Katalog danych o zmiennej, rozrastającej się strukturze atrybutów	Baza dokumentowa (tu: MongoDB)	Różne produkty mają różne atrybuty; sztywna tabela wymuszałaby puste kolumny albo ciągłe `ALTER TABLE`	Katalog produktów sklepu B2B na `mongo01`
Masowy zapis z wielu urządzeń naraz, proste zapytania po czasie i urządzeniu	Baza kolumnowa (tu: Cassandra)	Trzeba skalować zapis poziomo na wiele serwerów; złączeń między encjami praktycznie nie ma	Telemetria skanerów i wózków z `cass01` do `cass03`
Pilotaż dużego systemu klasy ERP z własnym silnikiem w pakiecie	Silnik wymagany przez ten system (tu: SAP HANA)	Jak w księgowości, ale to wciąż pilotaż: decyzja o pełnym wdrożeniu jeszcze nie zapadła	SID `ZHP` na `hana01`

Potrzeba	Silnik	Dlaczego	Przykład w Zielonej
Stary system, którego już nie rozwijamy, trzymany tylko do wglądu przez okres przechowywania	Silnik, na którym już stoi, bez migracji	Migracja archiwum kosztowałaby więcej niż jakakolwiek korzyść z jej wykonania	Archiwum płac `PLACE_ARCH` na `syb01`, tylko odczyt

Do rodzin silników z prawej kolumny tej tabeli (relacyjna, dokumentowa, kolumnowa) wraca osobno strona Relacyjne, dokumentowe, kolumnowe, klucz-wartość: mapa rodzin, z pełnym omówieniem mocnych stron i ceny każdej z nich.

Drzewo decyzji dla własnej, nowej aplikacji

Poniższe drzewo dotyczy wyłącznie sytuacji, w której naprawdę mamy wybór, czyli własnej, nowej aplikacji bez narzuconego z góry silnika. Pierwsze pytanie odsiewa właśnie te przypadki, w których wyboru nie ma.



Drzewo decyzyjne wyboru silnika bazy danych dla nowej potrzeby (kliknięcie otwiera pełny rozmiar)

Trzy pytania w środku drzewa (stała struktura ze złączeniami, zmienna struktura, masowy rozproszony zapis) odpowiadają dokładnie trzem pierwszym rodzinom ze strony o rodzinach baz. Czwartą gałąź, magazyn klucz-wartość, wybieramy rzadko jako główne, trwałe miejsce przechowywania danych; częściej ląduje obok głównej bazy jako warstwa cache, sesji albo kolejki, nie zamiast niej.

Koszty, kompetencje, wsparcie

Sama tabela technicznych mocnych stron nie wystarcza do decyzji, bo trzy inne czynniki bywają równie ważne, a rzadziej trafiają na pierwszą stronę dokumentacji technicznej.

- **Koszty licencji.** Silniki komercyjne (MSSQL, Oracle, SAP HANA, Sybase) liczą opłaty według modelu producenta, od liczby rdzeni procesora po liczbę użytkowników; silniki otwarte (PostgreSQL, MariaDB, MongoDB Community, Cassandra) nie mają opłaty za sam

silnik, ale nadal kosztują czas administratora. Konkretne stawki sprawdzamy każdorazowo u producenta albo u jego lokalnego partnera, nie tutaj.
- **Kompetencje zespołu.** Silnik, którego nikt w zespole nie zna, oznacza krzywą uczenia się doliczoną do każdego kolejnego zadania, nawet jeśli sam silnik pasuje do potrzeby lepiej na papierze. Zespół Oli wybrał PostgreSQL częściowo dlatego, że już go znał, i to jest w pełni uzasadniony powód wyboru, nie tylko wygoda.
- **Wsparcie.** Silnik komercyjny przychodzi zwykle z linią wsparcia producenta na czas awarii; silnik otwarty opiera się na dokumentacji, forum i własnej wiedzy zespołu, chyba że wykupimy wsparcie u firmy trzeciej. Dla systemu, od którego zależy zamknięcie miesiąca w księgowości, brak linii wsparcia bywa argumentem samym w sobie.

Kiedy nie potrzebujemy bazy wcale

Nie każda potrzeba przechowywania danych uzasadnia stawianie jakiegokolwiek silnika bazy danych. Kilkanaście wierszy danych, które zmienia jedna osoba, bez równoczesnego dostępu wielu użytkowników i bez potrzeby żadnego zapytania bardziej złożonego niż filtr w kolumnie, mieści się wygodnie w zwykłym pliku: arkuszu kalkulacyjnym, pliku CSV albo nawet pliku tekstowym. Baza danych zaczyna się opłacać dopiero tam, gdzie pojawia się więcej niż jedna z tych rzeczy naraz: kilka osób piszących jednocześnie, potrzeba pilnowania spójności między powiązanymi danymi, albo zapytania, na które arkusz nie odpowie bez ręcznego przeklikania każdego wiersza.

Częste błędy

Objaw	Przyczyna	Co robimy
Nowy wewnętrzny projekt startuje na silniku, którego nikt w zespole nie zna, bo „jest najnowocześniejszy"	Wybór podjęty pod wpływem mody albo jednej konferencyjnej prezentacji, bez sprawdzenia kompetencji zespołu	Wracamy do tabeli potrzeb: jeśli dane mają stałą strukturę i zespół zna PostgreSQL, nowość sama w sobie nie jest wystarczającym powodem zmiany
Dane katalogowe o zmiennej strukturze wciśnięte na siłę w sztywną tabelę relacyjną	Wybór silnika relacyjnego z przyzwyczajenia, bez sprawdzenia, czy dane w ogóle mają stałą strukturę	Rozważamy bazę dokumentową dla tego konkretnego zbioru danych, zamiast dokładać kolejne rzadko wypełnione kolumny do tabeli
Prosty arkusz z listą pięciu kontaktów zamieniony w projekt z własną bazą i aplikacją webową	Założenie, że „poważne dane" zawsze wymagają bazy danych, niezależnie od skali i liczby osób piszących naraz	Sprawdzamy sekcję „Kiedy nie potrzebujemy bazy wcale" wyżej, zanim zaczniemy projektować schemat dla czegoś, co zmieści się w jednym arkuszu

Objaw	Przyczyna	Co robimy
Decyzja o silniku dla nowej aplikacji zapada bez pytania o wsparcie na wypadek awarii w środku nocy	Skupienie wyłącznie na możliwościach technicznych silnika, pominięcie pytania „kto pomoże, gdy coś się zepsuje"	Przed wdrożeniem sprawdzamy, czy stawiamy na silnik z linią wsparcia producenta, czy wyłącznie na wiedzę własnego zespołu, i czy to świadoma decyzja

□□ Ćwiczenia

1. Nowy zespół w Zielonej chce zbudować wewnętrzny panel do rezerwacji sal spotkań: dane mają stałą strukturę (sala, godzina, osoba rezerwująca), a złączenia będą proste. Który silnik pasuje najlepiej i dlaczego, zgodnie z drzewem decyzji z tej strony?

Odpowiedź

Silnik relacyjny, tu PostgreSQL: dane mają stałą, powtarzalną strukturę i wymagają prostych złączeń (na przykład sala z jej rezerwacjami), a aplikacja nie przychodzi z żadnym narzuconym z zewnątrz silnikiem, więc pierwsze pytanie drzewa („aplikacja przychodzi z narzuconym silnikiem") od razu prowadzi do gałęzi „nie", a drugie pytanie („stała struktura, złączenia") do gałęzi „tak".

1. Dlaczego wybór MSSQL pod Optimę i wybór PostgreSQL pod aplikacje zespołu Oli nie są tą samą kategorią decyzji, mimo że oba dotyczą tego samego pytania „jaki silnik"?

Odpowiedź

Wybór MSSQL pod Optimę to w ogóle nie decyzja podjęta przez dział IT: silnik narzuciła licencja aplikacji kupionej od zewnętrznego producenta, więc jedyną realną decyzją było kupić Optimę albo nie. Wybór PostgreSQL pod własne aplikacje Oli jest prawdziwą decyzją inżynierską, opartą na strukturze danych, kompetencjach zespołu i braku przymusu licencyjnego, dokładnie tą, którą opisuje drzewo decyzji z tej strony.

1. Dział księgowości prosi o „bazę danych" do prowadzenia listy dwudziestu stałych dostawców, edytowanej wyłącznie przez panią Halinę. Czy potrzebują tu w ogóle silnika bazy danych?

Odpowiedź

Niekoniecznie: dwadzieścia wierszy edytowanych przez jedną osobę, bez równoczesnego zapisu wielu użytkowników i bez zapytań bardziej złożonych niż filtr, to dokładnie przypadek z sekcji

„Kiedy nie potrzebujemy bazy wcale" na tej stronie. Zwykły arkusz kalkulacyjny wystarczy, a stawianie pod to osobnego silnika bazy danych byłoby nakładem pracy nieproporcjonalnym do potrzeby.

Powiązane: Relacyjne, dokumentowe, kolumnowe, klucz-wartość: mapa rodzin · Bazy danych: od kart perforowanych do SQL · Baza danych, silnik, instancja, schemat

Źródła: dokumentacja PostgreSQL, rozdz. „What is PostgreSQL?" · dokumentacja MongoDB, rozdz. „When to Use MongoDB" · dokumentacja Apache Cassandra, rozdz. „Architecture" · sap.com, „SAP HANA overview"

Revision #21
Created 2026-09-28 17:51:24 UTC by Tomasz Konieczny
Updated 2026-10-26 14:10:43 UTC by Tomasz Konieczny