1. Czym jest SZBD / DBMS?

System Zarządzania Bazą Danych jako warstwa między użytkownikiem a danymi

- SZBD (DBMS) to specjalistyczne oprogramowanie służące do tworzenia, modyfikowania, wyszukiwania i zabezpieczania danych.
- Ukrywa szczegóły fizycznego zapisu danych na dysku i udostępnia użytkownikowi logiczny model bazy.
- Pośredniczy między aplikacjami, użytkownikami i plikami bazy danych.
- Pilnuje spójności, współbieżności, uprawnień oraz trwałości danych.
- Przykłady: PostgreSQL, MySQL, MariaDB, Oracle Database, Microsoft SQL Server, SQLite.

TRZY WARSTWY

Użytkownik / aplikacja
↓
SZBD / DBMS
↓
Pliki danych na dysku

PYTANIE KONTROLNE

Dlaczego aplikacja nie powinna samodzielnie zarządzać plikami bazy danych?

2. Baza danych a SZBD – to nie to samo

- Baza danych to uporządkowany zbiór danych zapisanych według określonego modelu.
- SZBD to program, który zarządza bazą i umożliwia wykonywanie na niej operacji.
- Jedno SZBD może obsługiwać wiele niezależnych baz danych.
- Usunięcie programu SZBD nie oznacza automatycznie usunięcia wszystkich danych, ale bez niego odczyt może być niemożliwy.
- W praktyce „serwer bazy danych” zwykle oznacza uruchomioną instancję SZBD.

Baza danych	Dane i struktura
SZBD	Oprogramowanie zarządzające
Instancja	Uruchomiony proces DBMS
Serwer	Komputer/usługa z DBMS

PYTANIE KONTROLNE

Czy plik .db lub zestaw plików danych sam w sobie jest SZBD?

3. Główne zadania systemu DBMS

- Definiowanie struktur danych: tabel, kolumn, typów, relacji i ograniczeń.
- Wprowadzanie, aktualizowanie, usuwanie i wyszukiwanie danych.
- Kontrola dostępu: użytkownicy, role, uprawnienia i uwierzytelnianie.
- Obsługa transakcji oraz równoczesnej pracy wielu użytkowników.
- Backup, odtwarzanie po awarii, logowanie i monitorowanie działania.

5 FUNKCJI DBMS

Struktura
Dane
Bezpieczeństwo
Transakcje
Odtwarzanie

PYTANIE KONTROLNE

Która funkcja DBMS odpowiada za zachowanie danych po awarii?

4. Architektura klient–serwer

- Klient wysyła zapytanie do serwera bazy danych przez sieć lub lokalne połączenie.
- Serwer DBMS interpretuje polecenie, planuje jego wykonanie i odczytuje odpowiednie strony danych.
- Wynik wraca do klienta jako zbiór rekordów, komunikat lub status operacji.
- Klientem może być aplikacja WWW, program desktopowy, skrypt lub narzędzie administracyjne.
- W architekturze wielowarstwowej aplikacja często korzysta z API, a nie łączy się z bazą bezpośrednio.

PYTANIE KONTROLNE

Który element wykonuje zapytanie SQL: klient czy serwer DBMS?

5. Warstwa logiczna i fizyczna danych

- Warstwa logiczna opisuje tabele, rekordy, kolumny, widoki i relacje widziane przez użytkownika.
- Warstwa fizyczna opisuje sposób zapisu stron, bloków, plików, indeksów i logów na nośniku.
- DBMS rozdziela te warstwy, dzięki czemu można zmieniać organizację zapisu bez modyfikowania aplikacji.
- To zjawisko nazywa się niezależnością danych.
- Dobra abstrakcja ogranicza zależność aplikacji od konkretnej organizacji plików.

ABSTRAKCJA

Tabela „uczniowie” jest pojęciem logicznym.
Na dysku może być zapisana w wielu plikach i blokach.

PYTANIE KONTROLNE

Co daje programiście niezależność danych od fizycznego sposobu zapisu?

6. Model relacyjny

- Dane są reprezentowane jako relacje, czyli w praktyce tabele.
- Wiersz reprezentuje pojedynczy rekord, a kolumna – atrybut.
- Każda kolumna ma określoną dziedzinę, najczęściej wyrażoną typem danych.
- Relacje między tabelami tworzy się z użyciem kluczy głównych i obcych.
- Operacje na danych opisuje algebra relacji, a w praktyce język SQL.

RELACJA

Tabela = relacja
Wiersz = krotka
Kolumna = atrybut
Typ = dziedzina

PYTANIE KONTROLNE

Jak w modelu relacyjnym nazywamy pojedynczy wiersz tabeli?

7. Tabela, rekord i kolumna

- Tabela przechowuje dane dotyczące jednego rodzaju obiektów, np. uczniów lub produktów.
- Rekord opisuje pojedynczy obiekt, np. jednego ucznia.
- Kolumna opisuje jedną cechę obiektu, np. nazwisko lub datę urodzenia.
- Nazwy kolumn powinny być jednoznaczne i technicznie poprawne.
- Każda kolumna powinna mieć dopasowany typ danych i – jeśli trzeba – ograniczenia.

id	101
imie	Anna
klasa	4B
aktywny	TRUE

PYTANIE KONTROLNE

Która część tabeli opisuje cechę obiektu: rekord czy kolumna?

8. Typy danych

- Typy liczbowe: INTEGER, BIGINT, NUMERIC/DECIMAL, REAL.
- Typy tekstowe: CHAR, VARCHAR, TEXT.
- Typy daty i czasu: DATE, TIME, TIMESTAMP.
- Typy logiczne: BOOLEAN.
- Współczesne DBMS oferują także JSON, UUID, dane binarne, tablice i typy przestrzenne.

DOBÓR TYPU

Typ danych wpływa na:
• poprawność
• zakres wartości
• miejsce na dysku
• szybkość operacji

PYTANIE KONTROLNE

Dlaczego numeru PESEL zwykle nie przechowuje się jako liczby?

9. Klucz główny (PRIMARY KEY)

- Klucz główny jednoznacznie identyfikuje każdy rekord w tabeli.
- Jego wartość nie może się powtarzać ani być NULL.
- Kluczem może być pojedyncza kolumna lub zestaw kilku kolumn.
- Często stosuje się techniczny identyfikator typu INTEGER lub UUID.
- PRIMARY KEY ułatwia tworzenie relacji i indeksowanie.

PRZYKŁAD

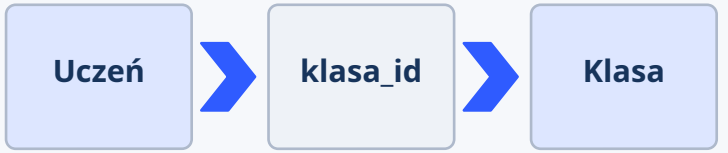
uczniowie
ID = 101 → Anna
ID = 102 → Jan
ID jest unikalny

PYTANIE KONTROLNE

Czy dwie osoby w jednej tabeli mogą mieć ten sam PRIMARY KEY?

10. Klucz obcy (FOREIGN KEY)

- Klucz obcy wskazuje rekord w innej tabeli i tworzy relację między tabelami.
- Najczęściej odwołuje się do PRIMARY KEY tabeli nadrzędnej.
- DBMS może wymuszać integralność referencyjną i blokować niepoprawne odwołania.
- Reakcje na usunięcie rekordu nadrzędnego można kontrolować przez CASCADE, RESTRICT, SET NULL itd.
- Klucze obce pomagają zachować spójność logiczną bazy.



PYTANIE KONTROLNE

Co się stanie, jeśli FOREIGN KEY wskazuje na rekord, który nie istnieje?

11. Ograniczenia integralności

- NOT NULL wymaga podania wartości.
- UNIQUE zabrania powtarzających się wartości.
- CHECK kontroluje warunek logiczny, np. cena > 0.
- DEFAULT nadaje wartość domyślną.
- PRIMARY KEY i FOREIGN KEY wymuszają integralność encji oraz referencji.

CEL

Złe dane powinny zostać odrzucone możliwie blisko źródła – najlepiej już przez DBMS.

PYTANIE KONTROLNE

Które ograniczenie zastosujesz, aby cena nie mogła być ujemna?

12. Język SQL – rola i zakres

- SQL jest deklaratywnym językiem komunikacji z relacyjnym DBMS.
- Użytkownik opisuje, jaki wynik chce uzyskać, a nie dokładny algorytm dostępu do danych.
- SQL służy do definiowania struktur, modyfikowania danych, odczytu i zarządzania uprawnieniami.
- Każdy producent DBMS może rozszerzać standard o własny dialekt.
- Podstawowe konstrukcje pozostają jednak bardzo podobne.

DEKLARATYWNOŚĆ

SQL: „pokaż uczniów z klasy 4B”
DBMS: sam wybiera sposób wykonania.

PYTANIE KONTROLNE

Dlaczego SQL nazywa się językiem deklaratywnym?

13. DDL – definicja danych

- DDL (Data Definition Language) opisuje strukturę bazy danych.
- CREATE tworzy obiekty, np. tabele, widoki i indeksy.
- ALTER modyfikuje strukturę istniejącego obiektu.
- DROP usuwa obiekt wraz z jego definicją.
- Zmiany DDL mogą wymagać szczególnej ostrożności w systemach produkcyjnych.

DDL

CREATE TABLE
ALTER TABLE
DROP TABLE
CREATE INDEX

PYTANIE KONTROLNE

Którym poleceniem SQL dodamy nową tabelę do bazy?

14. DML – modyfikowanie danych

- DML (Data Manipulation Language) służy do pracy na rekordach.
- INSERT dodaje nowe rekordy.
- UPDATE zmienia istniejące wartości.
- DELETE usuwa rekordy.
- Operacje DML zwykle uczestniczą w transakcjach i mogą być zatwierdzane lub wycofywane.

DML

INSERT
UPDATE
DELETE

PYTANIE KONTROLNE

Jaka jest różnica między DELETE a DROP TABLE?

15. SELECT – pobieranie danych

- SELECT pobiera dane z jednej lub wielu tabel, widoków albo innych źródeł.
- Lista SELECT określa, które kolumny lub wyrażenia mają znaleźć się w wyniku.
- FROM wskazuje źródła danych.
- WHERE filtruje rekordy przed zwróceniem wyniku.
- ORDER BY, LIMIT i DISTINCT pozwalają kontrolować końcową postać rezultatu.

SCHEMAT

```
SELECT kolumny
FROM tabela
WHERE warunek
ORDER BY ...
```

PYTANIE KONTROLNE

Która klauzula ogranicza rekordy na podstawie warunku?

16. Filtrowanie i operatory

- Porównania: =, <>, <, >, <=, >=.
- Operatory logiczne: AND, OR, NOT.
- BETWEEN sprawdza zakres, IN przynależność do zbioru, LIKE dopasowanie wzorca.
- NULL wymaga operatorów IS NULL lub IS NOT NULL.
- Warunki w WHERE powinny być tworzone tak, aby umożliwiały wykorzystanie indeksów.

NULL

NULL ≠ 0
NULL ≠ pusty tekst
NULL = brak znanej wartości

PYTANIE KONTROLNE

Dlaczego warunek „kolumna = NULL” jest niepoprawny?

17. Sortowanie i ograniczanie wyników

- ORDER BY porządkuje wynik rosnąco (ASC) lub malejąco (DESC).
- Można sortować po jednej lub wielu kolumnach.
- LIMIT/TOP/FETCH ogranicza liczbę zwracanych rekordów – składnia zależy od DBMS.
- Przy stronicowaniu wyników istotne jest deterministyczne sortowanie.
- Sortowanie dużych zbiorów może być kosztowne i wykorzystywać pamięć lub dysk.

PRZYKŁAD

```
ORDER BY nazwisko ASC, imie ASC
LIMIT 20
```

PYTANIE KONTROLNE

Co się może wydarzyć przy LIMIT 20 bez ORDER BY?

18. Funkcje agregujące

- COUNT liczy rekordy lub wartości nie-NULL.
- SUM sumuje wartości liczbowe.
- AVG oblicza średnią.
- MIN i MAX wyszukują wartości skrajne.
- Agregacje często współpracują z GROUP BY oraz HAVING.

AGREGACJA

1000 rekordów
↓
1 wynik lub kilka grup

PYTANIE KONTROLNE

Która funkcja zwróci liczbę uczniów w tabeli?

19. GROUP BY i HAVING

- GROUP BY dzieli rekordy na grupy według wskazanych kolumn.
- Funkcje agregujące są następnie liczone osobno dla każdej grupy.
- WHERE filtruje rekordy przed grupowaniem.
- HAVING filtruje już utworzone grupy.
- Poprawne użycie GROUP BY wymaga rozumienia kolejności logicznego wykonania zapytania.

KOLEJNOŚĆ

FROM → WHERE → GROUP BY → HAVING → SELECT → ORDER BY

PYTANIE KONTROLNE

Czym różni się WHERE od HAVING?

20. JOIN – łączenie tabel

- JOIN pozwala pobierać powiązane dane z wielu tabel.
- INNER JOIN zwraca tylko dopasowane rekordy.
- LEFT JOIN zachowuje wszystkie rekordy z lewej tabeli, nawet bez dopasowania.
- RIGHT JOIN i FULL JOIN są dostępne w części DBMS.
- Warunek ON powinien opisywać poprawne powiązanie między tabelami.

JOIN

```
uczniowie JOIN klasy
ON uczniowie.klasa_id = klasy.id
```

PYTANIE KONTROLNE

Który JOIN pokaże również uczniów bez przypisanej klasy?

21. Podzapytania i CTE

- Podzapytanie jest zapytaniem osadzonym wewnątrz innego zapytania.
- Może zwracać pojedynczą wartość, kolumnę, tabelę lub zbiór rekordów.
- CTE (WITH) tworzy nazwany tymczasowy wynik wykorzystywany w dalszej części zapytania.
- CTE często poprawia czytelność złożonych zapytań.
- Nie zawsze oznacza wzrost wydajności – optymalizacja zależy od DBMS.

CTE

```
WITH wynik AS (...)
SELECT *
FROM wynik;
```

PYTANIE KONTROLNE

Jaką główną zaletę daje CTE w porównaniu z bardzo zagnieżdżonym SQL?

22. Widoki (VIEW)

- Widok jest zapamiętaną definicją zapytania prezentowaną jak wirtualna tabela.
- Może upraszczać dostęp do złożonych danych.
- Pozwala ukrywać wybrane kolumny i zwiększać bezpieczeństwo.
- Zwykły widok najczęściej nie przechowuje własnej kopii danych.
- Widoki materializowane zapisują wynik i wymagają odświeżania.

VIEW

Dane fizyczne pozostają w tabelach.
Widok = logiczne „okno” na dane.

PYTANIE KONTROLNE

Czy zwykły widok przechowuje własną kopię wszystkich rekordów?

23. Indeksy – po co są?

- Indeks jest dodatkową strukturą danych przyspieszającą wyszukiwanie, sortowanie i łączenie.
- Działa podobnie do indeksu w książce: pomaga szybciej znaleźć właściwe miejsce.
- Najpopularniejszą strukturą indeksową jest B-tree.
- Indeks zajmuje miejsce i musi być aktualizowany podczas INSERT/UPDATE/DELETE.
- Zbyt wiele indeksów może spowalniać zapis.

KOMPROMIS

Więcej indeksów:
+ szybszy odczyt
– wolniejszy zapis
– więcej miejsca

PYTANIE KONTROLNE

Dlaczego nie należy tworzyć indeksu na każdej kolumnie?

24. Jak DBMS wykonuje zapytanie?

- Parser sprawdza składnię i buduje wewnętrzną reprezentację zapytania.
- Optymalizator analizuje możliwe strategie wykonania.
- Planista wybiera kolejność JOIN-ów, indeksy, skany i metody agregacji.
- Executor realizuje plan i zwraca wynik.
- Koszt planu jest estymowany na podstawie statystyk oraz modelu kosztowego.

PYTANIE KONTROLNE

Który moduł DBMS decyduje, czy użyć indeksu?

25. Plan wykonania zapytania

- Plan wykonania pokazuje, jak DBMS zamierza pobierać i łączyć dane.
- Typowe operacje: Seq Scan, Index Scan, Sort, Hash Join, Nested Loop, Aggregate.
- Koszt planu nie jest czasem w sekundach, lecz wewnętrzną estymacją.
- Analiza planu pomaga znajdować wąskie gardła i błędne indeksowanie.
- W wielu DBMS można porównać plan przewidywany z rzeczywistym wykonaniem.

DIAGNOSTYKA

Wolne zapytanie?
1. Sprawdź plan
2. Statystyki
3. Indeksy
4. Liczbę odczytów

PYTANIE KONTROLNE

Co analizujemy jako pierwsze przy bardzo wolnym SELECT?

26. Transakcje

- Transakcja to logiczna jednostka pracy składająca się z jednej lub wielu operacji.
- Zmiany mogą zostać zatwierdzone poleceniem COMMIT.
- W razie błędu można je wycofać poleceniem ROLLBACK.
- Transakcje chronią spójność podczas operacji wieloetapowych.
- Przykład: przelew bankowy powinien zmienić dwa salda jako jedną całość.

TRANSAKCJA

BEGIN
operacja 1
operacja 2
COMMIT / ROLLBACK

PYTANIE KONTROLNE

Dlaczego przelew bankowy nie powinien składać się z dwóch niezależnych UPDATE?

27. ACID

- Atomicity – transakcja wykonuje się w całości albo wcale.
- Consistency – baza przechodzi między stanami zgodnymi z regułami.
- Isolation – równoległe transakcje nie powinny zakłócać się w niedozwolony sposób.
- Durability – zatwierdzone dane przetrwają awarię.
- ACID jest fundamentem niezawodnych systemów transakcyjnych.

A	Atomicity
C	Consistency
I	Isolation
D	Durability

PYTANIE KONTROLNE

Która własność ACID oznacza trwałość zatwierdzonych zmian?

28. Współbieżność

- Współbieżność umożliwia wielu użytkownikom jednoczesną pracę na tej samej bazie.
- DBMS musi chronić dane przed konfliktami i niepoprawnymi odczytami.
- Wykorzystuje blokady, MVCC, znaczniki wersji i protokoły synchronizacji.
- Zbyt agresywne blokowanie zmniejsza wydajność, zbyt słabe – zwiększa ryzyko anomalii.
- Celem jest równowaga między poprawnością i przepustowością.

PROBLEM

Dwóch użytkowników edytuje ten sam rekord w tym samym czasie – DBMS musi ustalić bezpieczne zasady.

PYTANIE KONTROLNE

Po co DBMS stosuje mechanizmy współbieżności?

29. Poziomy izolacji transakcji

- Read Uncommitted – najsłabsza izolacja, dopuszcza wiele anomalii.
- Read Committed – każde zapytanie widzi tylko zatwierdzone dane.
- Repeatable Read – ponowny odczyt tych samych danych powinien być stabilniejszy.
- Serializable – najwyższa izolacja; efekt ma odpowiadać wykonaniu sekwencyjnemu.
- Wyższa izolacja zwykle zwiększa koszty i liczbę konfliktów.

ANOMALIE

Dirty read
Non-repeatable read
Phantom read
Write skew

PYTANIE KONTROLNE

Który poziom izolacji daje najsilniejsze gwarancje zgodności?

30. Blokady i zakleszczenia

- Blokada ogranicza jednoczesny dostęp do zasobu w celu ochrony spójności.
- Blokady mogą dotyczyć wierszy, stron, tabel lub innych obiektów.
- Deadlock powstaje, gdy transakcje wzajemnie czekają na zasoby.
- DBMS wykrywa zakleszczenie i zwykle przerywa jedną z transakcji.
- Pomaga krótki czas transakcji i stała kolejność dostępu do zasobów.

DEADLOCK

T1 trzyma A → czeka na B
T2 trzyma B → czeka na A

PYTANIE KONTROLNE

Jak DBMS najczęściej rozwiązuje wykryty deadlock?

31. MVCC – wielowersyjność danych

- MVCC (Multi-Version Concurrency Control) przechowuje lub śledzi wiele wersji rekordów.
- Odczyty mogą korzystać z właściwej wersji bez blokowania zwykłych zapisów.
- Transakcja widzi logiczny „snapshot” danych zgodny z regułami izolacji.
- Stare wersje muszą być okresowo usuwane lub porządkowane.
- MVCC jest powszechnie stosowane m.in. w PostgreSQL i Oracle.

IDEA MVCC

Czytelnik widzi wersję A
Pisarz tworzy wersję B
Obaj mogą pracować równocześnie

PYTANIE KONTROLNE

Jaka jest główna zaleta MVCC dla równoległych odczytów i zapisów?

32. Buforowanie i pamięć podręczna

- DBMS przechowuje często używane strony danych w pamięci RAM.
- Odczyt z RAM jest wielokrotnie szybszy niż odczyt z dysku.
- Buffer pool/cache zmniejsza liczbę fizycznych operacji I/O.
- DBMS stosuje algorytmy decydujące, które strony pozostają w pamięci.
- Rozmiar pamięci podręcznej silnie wpływa na wydajność serwera.

I/O

RAM → bardzo szybko
SSD → szybko
HDD → znacznie wolniej

PYTANIE KONTROLNE

Dlaczego duża pamięć RAM może przyspieszyć bazę nawet bez zmiany SQL?

33. Dziennik transakcyjny / WAL

- DBMS zapisuje informacje o zmianach w dzienniku przed zapisaniem właściwych stron danych.
- Technika Write-Ahead Logging pozwala odtworzyć spójny stan po awarii.
- Log pomaga zapewnić trwałość z ACID.
- Jest wykorzystywany także w replikacji i odzyskiwaniu punkt-w-czasie.
- Uszkodzenie lub brak logów może uniemożliwić pełne odtworzenie.

PYTANIE KONTROLNE

Dlaczego log zmian jest zapisywany przed właściwymi stronami danych?

34. Backup bazy danych

- Backup to kopia danych pozwalająca odtworzyć system po awarii, błędzie lub ataku.
- Backup logiczny zapisuje strukturę i dane w postaci poleceń/rekordów.
- Backup fizyczny kopiuje pliki lub bloki systemu bazodanowego.
- Kopie mogą być pełne, przyrostowe i różnicowe – zależnie od narzędzi.
- Kopia, której nigdy nie testowano przez odtworzenie, nie daje pewności działania.

ZASADA

Backup ≠ bezpieczeństwo
Dopiero test restore potwierdza, że kopia działa.

PYTANIE KONTROLNE

Dlaczego samo wykonanie backupu nie gwarantuje możliwości odtworzenia bazy?

35. Odtwarzanie po awarii

- Recovery ma przywrócić bazę do spójnego i użytecznego stanu.
- DBMS wykorzystuje backupy, logi transakcyjne i punkty kontrolne.
- RPO określa dopuszczalną utratę danych.
- RTO określa dopuszczalny czas niedostępności.
- Strategia odtwarzania powinna być częścią planu ciągłości działania.

RPO	Ile danych możemy stracić?
RTO	Jak długo system może nie działać?

PYTANIE KONTROLNE

Co opisuje RTO: utratę danych czy czas przywracania usługi?

36. Użytkownicy, role i uprawnienia

- DBMS identyfikuje użytkowników i procesy łączące się z bazą.
- Role grupują zestawy uprawnień i ułatwiają zarządzanie dostępem.
- Typowe prawa: SELECT, INSERT, UPDATE, DELETE, EXECUTE, CREATE.
- Zasada najmniejszych uprawnień oznacza nadawanie tylko tego, co konieczne.
- Konto aplikacji nie powinno mieć praw administratora bez wyraźnej potrzeby.

LEAST PRIVILEGE

Użytkownik powinien mieć dokładnie tyle praw, ile potrzebuje – ani mniej, ani więcej.

PYTANIE KONTROLNE

Dlaczego aplikacja WWW nie powinna logować się do bazy jako administrator?

37. Uwierzytelnianie i szyfrowanie

- Uwierzytelnianie potwierdza tożsamość użytkownika lub aplikacji.
- Możliwe mechanizmy: hasło, certyfikat, Kerberos, IAM, uwierzytelnianie systemowe.
- TLS szyfruje połączenie klient–serwer.
- Dane mogą być szyfrowane „w spoczynku” na dysku oraz w backupach.
- Bezpieczny system wymaga ochrony zarówno danych, jak i kluczy szyfrujących.

DWA KIERUNKI

In transit → TLS
At rest → szyfrowanie plików/nośników

PYTANIE KONTROLNE

Przed czym chroni TLS w połączeniu z bazą danych?

38. SQL Injection

- SQL Injection polega na wstrzyknięciu fragmentu SQL do danych wejściowych aplikacji.
- Najczęstszą przyczyną jest sklejanie zapytań z niezweryfikowanym tekstem użytkownika.
- Podstawową ochroną są zapytania parametryzowane / prepared statements.
- Dodatkowo stosuje się ograniczone uprawnienia konta aplikacji i walidację danych.
- „Escapowanie” tekstu nie powinno zastępować parametrów zapytania.

NIEBEZPIECZNE

SELECT … WHERE login = '${input}'

Bezpieczniej: parametr :login

PYTANIE KONTROLNE

Jaka technika jest podstawową ochroną przed SQL Injection?

39. Normalizacja danych

- Normalizacja porządkuje strukturę relacyjnej bazy i ogranicza redundancję.
- 1NF wymaga atomowych wartości i poprawnej struktury tabeli.
- 2NF usuwa zależności częściowe od złożonego klucza.
- 3NF ogranicza zależności przechodnie między atrybutami niekluczowymi.
- Normalizacja zmniejsza ryzyko anomalii wstawiania, aktualizacji i usuwania.

CEL

Mniej powtórzeń
Więcej spójności
Czytelniejsze zależności

PYTANIE KONTROLNE

Po co dzielimy powtarzające się dane na powiązane tabele?

40. Denormalizacja

- Denormalizacja celowo wprowadza redundancję, aby przyspieszyć wybrane odczyty.
- Może ograniczyć liczbę JOIN-ów i obliczeń podczas raportowania.
- Zwiększa jednak koszt aktualizacji oraz ryzyko niespójności.
- Powinna być stosowana świadomie, na podstawie pomiarów wydajności.
- W hurtowniach danych jest częstsza niż w klasycznych systemach OLTP.

KOMPROMIS

Normalizacja → spójność
Denormalizacja → czasem szybszy odczyt

PYTANIE KONTROLNE

Kiedy denormalizacja może być uzasadniona?

41. Replikacja

- Replikacja utrzymuje kopie danych na więcej niż jednym serwerze.
- Może zwiększać dostępność, odporność na awarie i skalować odczyty.
- Replikacja synchroniczna czeka na potwierdzenie z innych węzłów.
- Asynchroniczna jest szybsza, ale może powodować opóźnienie kopii.
- Replikacja nie zastępuje backupu – błąd logiczny może skopiować się na wszystkie repliki.

PYTANIE KONTROLNE

Dlaczego replikacja nie jest tym samym co backup?

42. Skalowanie baz danych

- Skalowanie pionowe oznacza mocniejszy serwer: więcej CPU, RAM i szybszy dysk.
- Skalowanie poziome oznacza dodawanie kolejnych serwerów lub węzłów.
- Repliki odczytowe rozkładają SELECT-y na wiele maszyn.
- Sharding dzieli dane na fragmenty przechowywane na różnych serwerach.
- Skalowanie poziome zwiększa złożoność aplikacji, monitoringu i spójności.

Vertical	Mocniejsza maszyna
Horizontal	Więcej maszyn
Replica	Więcej odczytów
Sharding	Podział danych

PYTANIE KONTROLNE

Na czym polega różnica między skalowaniem pionowym i poziomym?

43. DBMS relacyjny a NoSQL

- Relacyjne DBMS opierają się na tabelach, relacjach, schemacie i SQL.
- NoSQL obejmuje m.in. bazy dokumentowe, klucz-wartość, kolumnowe i grafowe.
- NoSQL bywa wybierane dla elastycznego schematu, ogromnej skali lub specyficznych modeli dostępu.
- Relacyjne DBMS nadal są bardzo silne w transakcjach, integralności i złożonych zapytaniach.
- Wybór technologii powinien wynikać z wymagań, a nie z mody.

RDBMS	Tabele, SQL, ACID
Dokument	JSON-like
Key-value	Klucz → wartość
Graf	Węzły i relacje

PYTANIE KONTROLNE

Czy NoSQL oznacza, że SQL i relacyjne bazy są przestarzałe?

44. Monitorowanie i administracja DBMS

- Administrator obserwuje obciążenie CPU, RAM, dysku i sieci.
- Kontroluje liczbę połączeń, blokady, deadlocki, cache hit ratio i czasy zapytań.
- Analizuje logi błędów i wolnych zapytań.
- Dba o aktualizacje, backupy, uprawnienia, pojemność i testy odtwarzania.
- Dobre monitorowanie pozwala wykryć problem zanim zauważy go użytkownik.

OBSERVABILITY

Metryki
Logi
Zapytania
Alerty
Trend pojemności

PYTANIE KONTROLNE

Jakie dwa symptomy mogą wskazywać na problem wydajnościowy DBMS?

45. Podsumowanie – jak myśli DBMS?

Od polecenia użytkownika do bezpiecznej i trwałej zmiany danych

- DBMS zarządza strukturą, danymi, dostępem, współbieżnością i trwałością.
- SQL opisuje zamiar użytkownika, a optymalizator wybiera sposób wykonania.
- Indeksy, cache i dobry plan zapytania decydują o wydajności.
- Transakcje, ACID, blokady i MVCC chronią spójność przy pracy wielu użytkowników.
- Backup, WAL, recovery, role, szyfrowanie i monitoring odpowiadają za niezawodność oraz bezpieczeństwo.
- Dobry projekt bazy łączy poprawność modelu danych z realnymi wymaganiami aplikacji.

ŁAŃCUCH DBMS

SQL → parser → optymalizator → transakcja → cache/dysk → log → wynik

PYTANIE KONTROLNE

Gdybyś miał wskazać trzy najważniejsze zadania DBMS, które byś wybrał i dlaczego?