1. System zarządzania bazami danych = kolekcja powiązanych ze sobą danych (baza danych) oraz programy, umożliwiające dostęp do tych danych
2. Cechy SZBD: efektywny, niezawodny, wygodny i bezpieczny dostęp wielu użytkowników do ogromnej ilości trwałych danych
3. Bazy danych tworzymy, by w sposób efektywny przechowywać duże ilości danych
4. Niedogodności plików BD: spora ilość programów, nadmiarowość i niespójność danych, izolacja danych, problemy z integralnością, hazard, bezpieczeństwo
5. Warstwa fizyczna BD to operacje na plikach i opis jak rekordy są przechowywane
6. Warstwa logiczna BD to operacje na tabelach i opis jakie dane i zależności
7. Warstwa widoku BD to ukrycie typów, ułatwienie interakcji, ukrywanie danych
8. Instancja to aktualna zawartość BD w określonym punkcie w czasie
9. Schemat to logiczna struktura BD: schemat logiczny (“zwykły”) i schemat fizyczny
10. Fizyczna niezależność - zdolność do modyfikowania schematu. fizycznego, bez modyfikacji logicznego; dane oddzielone od programu, nie są fizycznie jego częścią
11. Model danych to zbiór narzędzi konceptualnych do opisu danych, operacji, struktury, zależności, ograniczeń; umożliwia opisanie projektu na wszystkich poziomach
12. Przykłady modeli danych: relacyjny, związków encji, obiektowy, semistrukturalny
13. Model konceptualny to reprezentacja obiektów w uniwersalnym modelu, niezależnym od modelu implementacyjnego; jest podstawą schematu BD
14. Model implementacyjny jest wykorzystywany do implementacji modeli konceptualnych, odzwierciedla model konceptualny w konkretne struktury
15. Język definicji danych do DDL: CREATE, DROP, ALTER, ENABLE TRIGGER
16. Język manipulacji danych to DML: INSERT, UPDATE, DELETE
17. 2 typy DML: proceduralne DMLs (co i jak uzyskać), deklaratywne DMLs (co)
18. Funkcjonalne komponenty SZBD: menedżer składowania, przetwarzanie zapytań, menedżer transakcji
19. Menedżer składowania odpowiada za interakcje z menedżerem plików, efektywne składowanie, pobieranie, uaktualnianie w bazie danych
20. Menedżer składowania składa się z menedżera autoryzacji i integralności, menedżera plików, menedżera buforów, menedżera transakcji
21. Procesor przetwarzania zawiera interpreter DDL, kompilator DML, silnik wykonywania zapytań
22. Przetwarzanie zapytań to: zapytanie->parser->algebra->optymalizacja->plan działania->wykonanie->output
23. Architektura SZBD może być scentralizowana, równoległa (wieloprocesorowa architektura), rozproszona, działa jako klient-serwer
24. Aplikacje są zazwyczaj podzielone na dwie części (front-end, back-end), czasem na trzy (back-end rozbity na bazę i aplikację serwera)
25. W relacyjnym modelu danych struktury danych to relacje, operacje to selekcja, projekcja, połączenie, operacje na zbiorach, ograniczenia integralnościowe to klucze (primary i foreign), zawężenie, unikalność i wartość pusta
26. Relacja=tabela, krotka=wiersz, atrybut=kolumna
27. Cechy atrybutów to unikalna nazwa, zbiór wartości, atomowość, wartość NULL
28. Do łączenia dwóch tabel używa się klucza; może on być prosty lub złożony

29. Superklucz to kolumna lub zestaw kolumn jednoznacznie identyfikujący krotkę
30. Klucz kandydujący to superklucz o najmniejszej ilości atrybutów
31. Klucz główny to wybrany klucz kandydujący, nie może być NULL
32. Klucz obcy to atrybut, który wskazuje na klucz podstawowy
33. Integralność referencyjna jest zachowana, jeśli każda wartość klucza obcego jest równa jakiejś wartości klucza głównego w powiązanej tabeli nadrzędnej lub ewentualnie jest równa NULL
34. Wartość na którą wskazuje FK nie może być NULLem, ale FK może być NULLem
35. Zawężanie dziedziny to inaczej CHECK CONSTRAINTS
36. Algebra relacji jest proceduralnym językiem zapytań, składającym się ze zbioru operatorów, które biorą jedną lub dwie relacje i produkują nową jako wynik
37. Podstawowe operatory AR: selekcja (“where”): σ, projekcja(wybór kolumn): ∏, unia(“union”): ∪, różnica (“ale nie w…”) : – , produkt kartezjański(nauczyciel x uczniowie): x, przemianowanie(zmiana nazw): ρ, a dodatkowe operatory to: przecięcie zbiorów(jest i tu i tu): ∩, złączenie naturalne(łączy relacje po wspólnym atrybucie, przy złączeniu Theta raz tylko wyświetla kolumnę, którą złączył):⋈, przypisanie (zapisz zapytanie jako program sekwencyjny): ←
38. Operatory rozszerzonej algebry relacji to uogólniona projekcja (można używać operatorów arytmetycznych), operacje agregacji (avg, min, max, sum, count, eliminacja duplikatów)
39. Model związków encji ER to całościowa logiczna reprezentacja rzeczywistości - mapowanie znaczeń i zależności
40. Encja to rzeczywisty obiekt rozróżnialny od innych, posiadający atrybuty
41. Zbiór encji to zbiór obiektów tego samego typu, mającego te same atrybuty = “tabela”
42. Etapy projektowania ER: identyfikacja zbiorów encji, wybór odpowiednich atrybutów, ustalanie związków między zbiorami encji
43. Związek jest powiązaniem między kilkoma encjami
44. Zbiór związków jest matematyczną relacją między n>=2 encjami
45. Zbiory związków mogą mieć atrybuty
46. Encja=krotka, zbiór encji=tabela, związek = pomiędzy dwoma konkretnymi krotkami, zbiór związków = pomiędzy całymi tabelami
47. stopnie związków encji: unarny (1:1), binarny (M:N), ternarny (łączy 3 encje, np. kierowca otrzymuje mandat od policjanta za konkretne wykroczenie), n-arny
48. ograniczenie krotności to (1:1), (1:M), (M:1), (M:N)
49. Istnienie relacji: opcjonalne (ktoś może mieć konto lub nie), obowiązkowe (konto musi być ktosia)
50. Udział zbioru encji w zbiorze związków: całkowity/częściowy
51. Każdy zbiór encji może mieć dokładnie jeden klucz główny
52. Klucz główny w zbiorach związków wyznaczamy w przypadku 1:M tam, gdzie jest 1(chyba jednak M) , a w przypadku M:N kombinację kluczy głównych
53. Słaba encja to szczególny typ encji, których klucze składają się z atrybutów kluczowych innych encji (taka tabelka pośrednia wiele-do-wielu)
54. Zbiór encji nie posiadających klucza głównego określany jest jako zbiór słabych encji
55. Dyskryminator zbioru słabych encji to klucz częściowy zbioru encji, pozwalający na odróżnienie encji w zbiorze słabych encji

56. Agregacja to szczególny rodzaj powiązania, w którym można wyróżnić klasę całość oraz klasę część. Przykład: biblioteka – książka
57. Specjalizacja/generalizacja: klasa bardziej specyficzna, posiada wszystkie cechy klasy bardziej ogólnej. Przykład (k. ogólna – k. specyficzna): zwierzę – pies, narzędzie – młotek.
58. Specjalizację tworzymy za pomocą TOP-DOWN, Generalizację BOTTOM-UP
59. Ograniczenia na generalizację/specjalizację: zdefiniowane przez ograniczenie (wyżej jest jakiś atrybut, który to definiuje, np. student_type), zdefiniowane przez użytkownika (pracownicy przypisywani do zespołów, a zespołu do pracowników, nie ma automatycznego przypisania na zasadzie jakiegoś atrybutu)
60. Encje nie mogą należeć do więcej niż jednego zbioru encji niższego poziomu w pojedynczej generalizacji, gdy są ograniczenia są rozłączne (studenci do (under)/graduate ), a gdy są zachodzące to mogą do wielu (np. studenci do grup)
61. Ograniczenia kompletności: zupełne (student musi należeć do grupy) lub częściowe
62. Rodzaje atrybutów w ER to proste (niepodzielne), złożone (adres), jednowartościowe (ID), wielowartościowe (numery tel.), pochodne (wiek na podst. daty urodzenia)
63. Transformacja atrybutów związków encji na model relacyjny polega na zamienieniu atrybutów prostych na kolumny, rozłożeniu atrybutów złożonychi stworzeniu tabel, wielowartościowych na tabele z kluczem głównym, pominięciu pochodnych
64. Zamiana związków ternarnych na binarne polega na stworzeniu sztucznego zbioru encji, np. w przykładzie z mandatami stworzeniu tabeli z trzema kluczami obcymi
65. Widok to dowolna relacja, która nie jest częścią modelu konceptualnego, ale jest widoczna dla użytkownika jako “wirtualna relacja”
66. Zalety widoków to m. in. ułatwienie w pisaniu zapytań, uprawnieniach, ukrywają złożoność bazy, pokazują tylko właściwe dane
67. Rodzaje widoków to zmaterializowane (stworzenie fizycznej relacji) i niezmaterializowane
68. Widoki zmaterializowane tworzymy przez CREATE MATERIALIZED VIEW
69. Dane w widoku zmaterializowanym odświeżamy albo całkowicie (czyli wszystko od nowa) albo przyrostowo (tylko o to, co dodajemy)
70. Możemy odświeżać natychmiast po zmianach lub leniwie, dopiero gdy potrzebujemy
71. Nie należy wstawiać do widoków, chyba że nie aliasujemy, nie mamy nulli, joinów, group by i having; ale i tak będzie to raczej mało optymalne
72. Kiedy stworzymy widok z WITH CHECK OPTION to wtedy nie możemy wstawić do widoku, jeśli rekord nie będzie spełniał WHERE
73. Indeks to struktura danych, która przyspiesza dostęp do rekordów o określonych wartościach dla atrybutów indeksów
74. Indeksy są trochę jak spis treści; są używane w instrukcjach SELECT
75. Warunki integralnościowe zabezpieczają bazę przed uszkodzeniem
76. Warunki integralnościowe możemy definiować dla pojedynczego atrybutu i całej relacji; dzielą się na notnull, PK, FK, unique, CHECK
77. Przy dodawaniu nowego constrainta do istniejącej tabeli, SZBD automatycznie sprawdza, czy istniejące dane spełniają warunki constrainta
78. Można blokować contrainty ze względów wydajnościowych, przy zmianie warunków
79. Blokujemy też na istniejących, gdy dane nie ulegają zmianie; możemy blokować CHECK i FK, resztę trzeba usuwać i tworzyć od nowa

80. Integralność referencyjna pomaga zapewnić poprawność danych, pozwala uniknąć usunięcia powiązanych danych
81. Przy tworzeniu tabeli można zdefiniować klucz obcy jako "on delete cascade". W momencie usunięcia krotki, która była kluczem obcym dla innej z klauzulą "On delete cascade" ta krotka jest usuwana automatycznie.
82. Usunięcie wiersza tabeli klucza głównego jest zabronione, dopóki nie zostaną usunięte (lub zmodyfikowane) wiersze z tabel, których wartości kluczy obcych stałyby się wskutek tej operacji nieaktualne
83. Usuwanie kaskadowe powoduje, że usunięcie wiersza z tabeli po stronie klucza głównego powoduje automatyczne usunięcie z tabel powiązanych wszystkich wierszy, dla których wartości kluczy obcych stałyby się nieaktualne.
84. Klauzula WITH pozwala nadać nazwę podzapytaniu
85. Procedura skalarna to procedura zwracająca skalar
86. Podzapytanie skalarne to podzapytanie zwracające jeden atrybut
87. Procedura składowana pozwala na czytelniejsze komunikowanie się klienta z bazą, ułatwia sprawę uprawnień
88. Duże obiekty są przechowywane jako blob lub clob
89. Blob (Binary Large OBject) umożliwia przechowywanie dużych ilości danych binarnych jako pojedynczy obiekt, stosowany w szczególności do przechowywania danych multimedialnych; nie są wyświetlane w SELECT
90. Clob (Character Large Object) -posiadają określone kodowanie znaków
91. W przypadku krotek wynikowych zawierających duże obiekty (od kilku megabajtów do gigabajtów), odzyskiwanie całego dużego obiektu do pamięci jest nieefektywne lub niepraktyczne. Zamiast tego, aplikacja zwykle używa zapytania SQL, by pobrać "lokalizator" dla dużego obiektu, lokalizator może być następnie użyty do pobrania dużego obiektu w małych kawałkach, a nie wszystkich naraz
92. Typy własne - distinct types pomagają w znajdowaniu błędów, bo jeśli stworzymy zmienne dollar i złoty to jak będziemy chcieli jedno wsadzić do drugiego to błąd
93. Kiedyś można było tworzyć dziedziny, ale one nie były silnie typowane i można na nie nakładać ograniczenia, typu NOT NULL czy wartości domyślne
94. Przy złożonych zapytaniach wygodnie jest przechowywać wynik zapytania jako nową tabelę (tabelę chwilową [temporary]): CREATE TABLE temp AS (SELECT ...) with data; (bez “with data” tabela nie zostanie wypełniona danymi)
95. Rozszerzenie tabeli trzyma nam tabelę w danym momencie, widoki zmaterializowane możemy aktualizować, widoki niezmaterializowane są zawsze aktualne
96. Role definiujemy przez CREATE ROLE i służą one do autoryzowania użytkowników do wykonywania pewnych operacji w bazie
97. Uprawnienia do modyfikacji schematu ma tylko właściciel (nie administrator!)
98. Formy autoryzacji do modyfikowania schematu bazy to Index (indeksy), Resources(tworzenie relacji), Alteration(atrybuty), Drop (usuwanie relacji)
99. By nadać uprawnienia używamy “GRANT references ON <table> TO <grupa użytkowników>”, by nadać wszystkie “GRANT all privileges”
100. grant reference (dn) on dep to Jon - pozwala Jonowi na tworzenie relacji, które odnoszą się do klucza dn tabeli dep jako klucz obcy
101. Trzy formy SQL: Interakcyjny, Statyczny (osadzony, język modułów) i Dynamiczny

102. Osadzony SQL oznacza włączenie kodu SQL do kodu źródłowego innego języka
103. Cechy osadzonego SQL to wyrażenie EXEC SQL jako identyfikator, przed wykonaniem wyrażenia, program musi się połączyć z bazą danych, mogą być używane zmienne języka macierzystego (deklarowane w DECLARE), nie można bezpośrednio osadzić SELECT-FROM-WHERE, zapytania są wysyłane w momencie kompilacji
104. Mechanizmy łączenia wyników zapytań z programem w języku macierzystym to jednowierszowe SELECT i kursory (declare c cursor for <SQL query>
105. Instrukcje osadzonego SQL to: open, fetch (zwraca 1. wiersz) i close
106. Dynamiczny SQL generowany jest w trakcie pracy aplikacji, kiedy nie znamy jeszcze treści zapytań; powstaje w oparciu o decyzje użytkownika
107. Dwa główne standardy dynamicznego SQL to JDBC i ODBC
108. JDBC (Java Database Connectivity) obsługuje wydawanie zapytań i modyfikację danych oraz pobieranie wyników zapytań; obsługuje pobieranie metadanych
109. ODBC (Open Database Connectivity) otwiera połączenia z bazą danych, przesyła zapytania i modyfikacje, otrzymuje wyniki
110. Funkcje tablicowe dają w wyniku sparametryzowane widoki (tablice)
111. Persistent storage module (PSM) pozwala na deklarowanie zmiennych (DECLARE), wyrażenia złożone (BEGIN...END), wyrażenia (REPEAT, WHILE)
112. CASE rozpatruje przypadki, w każdym przypadku należy zdefiniować warunek
113. Funkcje zewnętrzne to funkcje napisane w innych językach (Java, C# itd.)
114. Korzyści z funkcji zewnętrznych to efektywność dla wielu operacji i uniwersalność, lecz wady to podatność na błędy i problemy z bezpieczeństwem
115. Rozwiązywanie problemów bezpieczeństwa przy zewnętrznych procedurach polega albo na “sandbox’ (procedury mają własną pamięć, ale nie mają dostępu do pamięci procesów i plików BD) albo uruchomienie zewnętrznych funkcji/procedur w osobnym procesie
116. Trigger to wyrażenie wykonywane automatycznie po modyfikacji BD
117. Triggery są, by implementować warunki integralnościowe, by uruchamiać zadania
118. Triggery nie mogą wykonywać modyfikacji poza bazą danych
119. Typy triggerów: BEFORE (przed), AFTER(po), INSTEAD OF(zamiast)
120. Zdarzenia powodujące wykonanie triggerów: INSERT, UPDATE, DELETE
121. Użycie klauzuli FOR EACH ROW oznacza typ wyzwalacza na poziomie wierszy; w innym przypadku jest to wyzwalacz poziomu instrukcji
122. Wyzwalacze na poziomie polecenia wykonują się przed lub po całym poleceniu, niezależnie od modyfikacji
123. Nie używamy triggerów, gdy zarządzamy danymi sumarycznymi, replikujemy BD, kaskadowo usuwamy, gdy możemy to rozwiązać przez CHECK CONTRAINTS
124. Problemy z triggerami to wydajność, złożoność, nie można planować kolejności
125. Relacyjną bazę danych projektujemy, najpierw zbierając wymagania użytkownika, opracowując strukturę, wybierając model, tłumacząc na konceptualny model, projektując logiczny i fizyczny projekt
126. Anomalie to tracenie informacji, podatność na brak spójności lub niemożliwość wykonania operacji na danych spowodowane złym schematem BD; rodzaje to redundancja, anomalia wprowadzania/usuwania/aktualizacji danych

127. Zależność funkcyjna zachodzi wtedy, gdy w ramach krotek relacji r wartości atrybutów zbioru X determinują jednoznacznie wartości atrybutów zbioru Y
128. Zależności funkcyjne możemy wykorzystać do wykrywania błędów w projekcie ER, sprawdzenia czy instancje relacji spełniają dany zbiór F zależności funkcyjnych
129. Wielowartościowe zależności funkcyjne możemy wykorzystywać do 4NF, testów
130. Legalna relacja to taka, która spełnia wszystkie nałożone zależności funkcyjne
131. Zbiór wszystkich zależności funkcyjnych logicznie wynikających ze zbioru F nazywany jest domknięciem zbioru F (oznaczane F+)
132. Domknięcie zbioru zależności wyznaczamy przez Aksjomat Amstronga lub przez użycie domknięcia atrybutów
133. Nadklucz: wartość każdego atrybutu ma być jednoznacznie zdeterminowana przez wartości atrybutów zbioru X. Jednym z nadkluczy jest zawsze zbiór wszystkich atrybutów R
134. Kluczem relacji nazywamy każdy minimalny nadklucz (nie zawierający w sobie żadnego innego nadklucza)
135. Aksjomaty Armstronga: reguła zwrotności ( if $\beta \subseteq \alpha$, then $\alpha \rightarrow \beta$), rozszerzalności ( if $\alpha \rightarrow \beta$, then $\gamma\alpha \rightarrow \gamma\beta$ ), przechodniości (if $\alpha \rightarrow \beta$, and $\beta \rightarrow \gamma$, then $\alpha \rightarrow \gamma$); ($\gamma$ to nowy zbiór, a $\gamma\alpha$ to suma zbiorów $\gamma$ i $\alpha$); reguły te są pewne i kompletne;
136. Reguły, które można wyprowadzić z poprzednich to Unia ( if $\alpha \rightarrow \beta$ and $\alpha \rightarrow \gamma$ , then $\alpha \rightarrow \beta\ \gamma$), dekompozycja (if $\alpha \rightarrow \beta\ \gamma$, then $\alpha \rightarrow \beta$ i $\alpha \rightarrow \gamma$), pseudoprzechodniość (if $\alpha \rightarrow \beta$ and $\gamma\ \beta \rightarrow \delta$, then $\alpha\ \gamma \rightarrow \delta$)
137. Algorytm wyliczania F+: powtarzaj dla każdej zależności funkcyjnej f w wyniku: zastosuj zasady zwrotności i rozszerzania na f, dodaj wynikowe zależności funkcyjne do wyniku, dla każdej pary zależności funkcyjnych f1 i f2 w wyniku: , jeśli f1 i f2 można połączyć zasadą przechodniości , to dodaj wynikową zależność funkcyjną do wyniku, dopóki (ostatnia iteracja nie zmieniła wyniku) , zwróć wynik (domknięcie)
138. 1NF: relacja opisuje 1 obiekt, wartości atrybutów są atomowe, nie zawiera kolekcji, wszystkie atrybuty niekluczowe są w zależności funkcyjnej od klucza
139. 2NF: 1NF + żadna kolumna nie jest funkcyjnie zależna od pewnej części klucza potencjalnego
140. 3NF: jeżeli dla wszystkich zależności funkcyjnych $\alpha \rightarrow \beta$ w F+ ($\alpha \subseteq R$, $\beta \subseteq R$) zachodzi przynajmniej jeden warunek: $\alpha \rightarrow \beta$ jest trywialną zależnością lub $\alpha$ jest superkluczem w R lub każdy atrybut A w $\beta$ - $\alpha$ znajduje się w kluczu kandydującym dla R
141. 3NF: 2NF + żaden atrybut niekluczowy nie jest zależny funkcyjnie od innych atrybutów niekluczowych
142. Zawsze możliwe jest zdekomponowanie schematu do 3NF w sposób bezstratny z zachowaniem wszystkich zależności, jednak czasem trzeba użyć NULLi (BCNF nie)
143. BCNF: jeżeli dla wszystkich zależności funkcyjnych $\alpha \rightarrow \beta$ w F+ ($\alpha \subseteq R$, $\beta \subseteq R$) zachodzi przynajmniej jeden warunek: $\alpha$ jest superkluczem w R lub $\alpha \rightarrow \beta$ jest trywialną zależnością
144. Żeby przekształcić relację do BCNF należy dokonać rekompozycji na szereg relacji, spełniających BCNF
145. Nie każdy schemat tabeli da się sprowadzić do zbioru schematów tabel w postaci normalnej Boyce'a-Codda - bez utraty zawartych w tabelach informacji i z zachowaniem zależności funkcyjnych

146. Domknięcie zbioru atrybutów a nad F (ozn. a+) to zbiór atrybutów, które są funkcyjnie określone przez a nad F
147. Zastosowania a+ to testowanie superklucza (dla domniemanego superklucza wyliczamy domknięcie - jeżeli jest nim cała tabela, to mamy superklucz) i zależności funkcyjnych (Sprawdzamy czy $\alpha \rightarrow \beta$ zachodzi przez sprawdzenie czy $\beta \subseteq \alpha+$) i alternatywne wyliczanie domknięcia F+
148. 8.16 i 8.17, 8.18, 8.19, TODO
149. Dekompozycja relacji to rozbicie jednej relacji na dwie lub więcej relacji
150. Cechy dobrej dekompozycji to eliminowanie anomalii, odtwarzalność informacji, zachowanie zależności
151. Jeśli R1 ∩ R2 tworzy superklucz w R1 lub R2 to dekompozycja R jest bezstratna
152. Dekompozycja zachowuje zależność jeśli spełnia warunek ( F1u F2 u ... u Fn )+ = F+ ; gdzie Fi to restrykcja z F nad Ri. F i jest podzbiorem takich zależności z F+, które zawierają tylko atrybuty z Ri.
153. 4NF: BCNF + nie ma zależności wielowartościowych
154. Wielowartościowe zależności wykorzystujemy do sprawdzania relacji w celu ustalenia czy są legalne dla zbioru zależności i określenia ograniczeń na zbiorze legalnych relacji
155. Dane czasowe mają skończony przedział czasowy, w którym są ważne
156. Klucz główny powinien być stały w czasie, klucz obcy powinien mieć referencję do aktualnej wersji danych lub do danych w konkretnym punkcie czasowym
157. PCTFREE liczba – określa, jaka część bloku jest zarezerwowana na rozrost wierszy, które już są w bloku danych (w % ) – domyślną wartością jest PCTFREE 10
158. PCTUSED liczba – określa rozmiar przestrzeni bloku, poniżej którego będzie dozwolone wstawianie nowego rekordu (tj. określa poziom graniczny, przy którym blok może trafić z powrotem na listę wolnych bloków) (wyrażony w % ) – domyślną wartością jest PCTUSED 40
159. Freelist - dzięki niej możemy dodawać nowe rekordy w miejsce usuniętych
160. Dla rekordów o stałej długości możemy przechowywać rekordy zaczynając od bajtu n*(i-1), ale są wtedy trudności przy usuwaniu
161. w pamięci rekordy o atrybutach o stałej i zmiennej długości reprezentowane są w taki sposób: najpierw pary(offset, dlugosc zmiennego atrybutu) potem stałej długości, null bitmapa, i potem zmiennej długości.
162. Strukruktura slotowa Slotted page jest przeważnie używana do przechowywania rekordów o zmiennej długości w bloku
163. Nagłówek Slotted Page - Block Header zawiera liczbę rekordów, koniec wolnej przestrzeni i tablicę z informacją o lokalizacji i rozmiarze każdego rekordu
164. Po nagłówku jest wolna przestrzeń, a po niej rekordy
165. Przy dodawaniu rekordu alokowana jest przestrzeń i dodawane info do tabeli (lokalizacja i rozmiar)
166. Przy usuwaniu rekordu przestrzeń jest zwalniana, rekordy wyżej są przesuwane “do dołu”, aktualizowany jest wskaźnik na koniec wolnej przestrzeni
167. Nie ma konieczności przechowywania wskaźników bezpośrednio do rekordów - zamiast tego wskaźniki wskazują na nagłówek każdego bloku, a nagłowek każdego bloku na konkretne rekordy. Pozwala to na przesuwanie rekordów bez fragmentacji.

168. Rekordy zmiennej długości istnieją, by zaoszczędzić miejsce; w innym wypadku stała długość musiałaby być wyznaczona przez najdłuższy element
169. Organizacja rekordów w plikach może być nieuporządkowana, uporządkowana, haszkowana lub klastrowana wielotablicowo
170. Organizacja uporządkowana jest przy uporządkowaniu sekwencyjnym, w zależności od wartości klucza każdego rekordu; aby umożliwić szybkie pobieranie, łączymy rekordy według wskaźników, gdzie każdy rekord wskazuje na kolejny
171. Gdy wstawiamy do pliku sekwencyjnego trzeba zlokalizować rekord, który jest przed wstawianym i jeśli jest tam pusta przestrzeń to wstaw, jeśli nie to mamy overflow block, na końcu trzeba dostosować wskaźniki
172. Kiedy mamy za dużo overflow blocków, powinniśmy reorganizować; częstość reorganizacji zależy od tego jak często wstawiamy
173. Przy organizacji przez haszowanie rezultat funkcji haszującej definiuje blok
174. Przy organizacji przez klastrowanie wielotablicowe BD przechowuje powiązane zapisy dwóch lub więcej relacji w każdym bloku. Taka organizacja plików pozwala nam odczytywać rekordy, które spełniają warunek łączenia przy użyciu odczytu jednego bloku; dzięki temu jesteśmy w stanie wydajniej przetworzyć to zapytanie
175. Słownik danych (katalog systemowy) przechowuje metadane; czyli dane o danych, takie jak: informacje o relacjach, użytkownikach, o fizycznej organizacji plików, o indeksach, dane statystyczne i opisowe
176. Słownik używany jest zarówno przez SZBD jak i użytkowników
177. Fizyczna organizacja słownika to na ogół nieznormalizowana minibaza
178. Menedżer bufforu - podsystem odpowiedzialny za alokowanie bufora do pamięci głównej. koniecznie wspomnieć, że on zapisuje z bufora na dysk
179. Menedżer bloków danych buforuje bloki danych, czyli sprawia by jak najwięcej bloków mieściło się w pamięci operacyjnej i minimalizuje ilość przesyłanych danych z plików do bufora
180. 9.13 TODO
181. Transakcja to sekwencja logicznie powiązanych operacji na bazie danych, która przeprowadza bazę danych z jednego stanu spójnego w drugi stan spójny
182. Transakcje są NIST: niepodzielne,  izolowane, spójne, trwałe
183. Transakcja może zaczynać się jawnie (BEGIN TRANSACTION) lub niejawnie i kończyć jako commit work lub rollback work
184. Każdy etap transakcji jest logowany, dzięki czemu w razie awarii systemu (dzięki zawartości logów) można odtworzyć stan bazy danych sprzed transakcji, która nie została zamknięta
185. Diagram stanów transakcji: 1. aktywna > częściowo wypełniona > wypełniona>zakończona, 2. aktywna > nieudana > zakończona, 3. aktywna > częściowo wypełniona > nieudana > zakończona
186. Wielodostęp - pogodzenie działań wielu transakcji. Instrukcje dwóch transakcji przeplatają się i transakcje są wykonywane jednocześnie. Nawet jeżeli każda z dwóch transakcji jest poprawna, przemieszanie ich operacji może spowodować wystąpienie błędów w bazie
187. Problemy wielodostępu: utrata zmian, niezatwierdzenie zależności, analiza niespójności

188. Fantom to wiersz, który zostaje wstawiony do tabeli po tym jak transakcja wykonała operację na tej tabeli a przed jej zatwierdzeniem
189. Szeregowalność - pozwala poznać wyniki transakcji, które na pewno nie zaburzą spójności bazy; rozpoznaje harmonogramy niesekwencyjne, które pozwalają wykonywać transakcje współbieżnie, nie dopuszczając do niepożądanych oddziaływań między nimi
190. Harmonogram to ciąg operacji wykonywanych sekwencyjnie , w których zachowany jest wewnętrzny porządek każdej transakcji
191. Harmonogram sekwencyjny składa się z ciągu akcji transakcji, które się nie przeplatają, tzn. najpierw wykonywana jest cała jedna transakcja, potem druga itd.
192. Harmonogram niesekwencyjny zawiera przeplatające się elementy obu transakcji, ale kolejność w obrębie transakcji jest zachowana.
193. Harmonogram szeregowalny doprowadza do takiego samego stanu bazy danych, jak pewne wykonanie sekwencyjne tych transakcji; zachowuje spójność bazy danych, przy założeniu, że wszystkie transakcje zakończą się powodzeniem
194. Harmonogram odtwarzalny: jeżeli żadna transakcja T w harmonogramie S nie jest zatwierdzona do momentu, aż wszystkie transakcje T', które zapisały element odczytywany przez transakcji T, zostaną zatwierdzone
195. Graf pierwszeństwa pozwala wykryć naruszenie szeregowalności; to graf skierowany;  jeśli w grafie istnieje krawędź Ti>Tj to w każdym równoważnym grafie Ti występuje przed Tj, no i Ti musi wykonać się przed Tj
196. 10.8 10.9 TODO
197. Dla zapewnienia szeregowalności zapobiegającej niespójności wynikającej z oddziaływań pomiędzy wykonywanymi jednocześnie transakcjami, ważne jest wzajemne uporządkowanie operacji odczytu i zapisu tych transakcji
198. Jeżeli dwie transakcje jedynie czytają ten sam element danych, to ich wzajemna kolejność nie jest istotna
199. Jeżeli dwie transakcje czytają lub zapisują różne elementy danych, to nie kolidują ze sobą i ich wzajemna kolejność nie jest istotna
200. Jeżeli jedna transakcja zapisuje element danych, a druga go odczytuje lub zapisuje, to kolejność wykonania tych transakcji ma znaczenie
201. Harmonogramy równoważne różnią się kolejnością poszczególnych operacji, ale przekształcające bazę do takiego samego stanu
202. Szeregowalność można zapewnić na kilka sposobów: Metody optymistyczne i metody pesymistyczne
203. Metody pesymistyczne to blokady i znaczniki czasowe
204. Metoda oparta na blokadach polega na tym, że transakcja, która odwołuje się do danych blokuje innym transakcjom dostęp do danych
205. Blokady dzielą się na dzielone/read lock (odczytuje, ale nie zmienia) i wyłączne/write lock (i odczytuje i zmienia)
206. Rozszerzanie blokady to zamiana z dzielonej na wyłączną, a redukcja na odwrót
207. Transakcja przestrzega protokołu blokowania dwufazowego (2PL) jeżeli wszystkie operacje zakładania blokady znajdują się w niej przed pierwszą operacją zdjęcia blokady
208. Protokół Strict-2PL nie dopuszcza do powstania anomalii niezatwierdzonego odczytu, niepowtarzalnego odczytu i nadpisywania nie zatwierdzonych danych

209. Jeżeli wszystkie transakcje spełniają protokół dwufazowego blokowania, to wszystkie harmonogramy niesekwencyjne są szeregowalne
210. Do tworzenia harmonogramu wykorzystuje się znaczniki czasowe (timestamps), które oznaczają czas wejścia transakcji w trzy fazy: Start, Validation, Finish
211. Metoda optymistyczna to walidacja
212. Gdy transakcja wykonuje test walidacji, to sprawdza, czy zmienne lokalne mogą być zapisane bez naruszenia integralności BD;  jeśli walidacja przebiegła pomyślnie zmienne lokalne są zapisywane w BD, w przeciwnym wypadku następuje rollback