1.

Postulat informacyjny - dane są reprezentowane jedynie poprzez wartości atrybutów w wierszach tabel,

2.

Postulat dostępu - każda wartość w bazie danych jest dostępna poprzez podanie nazwy tabeli, atrybutu oraz wartości klucza podstawowego,

3.

Postulat dotyczący wartości NULL - dostępna jest specjalna wartość NULL dla reprezentacji wartości nieokreślonej jak i nieadekwatnej, inna od wszystkich i podlegająca przetwarzaniu

4.

Postulat dotyczący katalogu - wymaga się, aby system obsługiwał wbudowany katalog relacyjny z bieżącym dostępem dla uprawnionych użytkowników używających języka zapytań,

5.

Postulat języka danych - system musi dostarczać pełnego języka przetwarzania danych, który może być używany w trybie interaktywnym jak i w obrębie programów aplikacyjnych, obsługuje operacje definiowania danych, operacje manipulowania danymi, ograniczenia związane z bezpieczeństwem i integralnością oraz operacje zarządzania transakcjami,

6.

Postulat modyfikowalności perspektyw - system musi umożliwiać modyfikowanie perspektyw, o ile jest ono (modyfikowanie) semantycznie realizowalne,

7.

Postulat modyfikowalności danych - system musi umożliwiać operacje modyfikacji danych, musi obsługiwać operatory INSERT, UPDATE oraz DELETE,

8.

Postulat fizycznej niezależności danych - zmiany fizycznej reprezentacji danych i organizacji dostępu nie wpływają na aplikacje,

9.

Postulat logicznej niezależności danych - zmiany wartości w tabelach nie wpływają na aplikacje,

10.

Postulat niezależności więzów spójności - więzy spójności są definiowane w bazie i nie zależą od aplikacji,

11.

Postulat niezależności dystrybucyjnej - działanie aplikacji nie zależy od modyfikacji i dystrybucji bazy,

12.

Postulat bezpieczeństwa względem operacji niskiego poziomu - operacje niskiego poziomu nie mogą naruszać modelu relacyjnego i więzów spójności.

Model hierarchiczny:
Dane układane są hierarchicznie, w strukturę drzewa uporządkowanego od ogółu do szczegółu. Model ten do nawigacji wśród przechowywanych danych wykorzystuje wskaźniki.

• Przy aktualizacji danych koniecznie jest aktualizowanie tych danych zawsze w wielu miejscach w celu uniknięcia powstawania niespójności danych.
• Tabela nadrzędna może posiadać wiele tabel podrzędnych. • Tabela podrzędna może mieć tylko jedną tabel nadrzędna.
• W celu odczytania danych z tabeli podrzędnej trzeba najpierw odczytać tabelę nadrzędną.

Model sieciowy:
tak jak model hierarchiczny, wykorzystuje on wskaźniki wskazujące przechowywane dane. Nie musi jednak korzystać ze struktury drzewa.

Rozwój modelu hierarchicznego.
• Tabele podrzędne mogą mieć wiele tabel nadrzędnych.
• Nie ma konieczności odczytywanie głównej tabeli w celu uzyskania dostępu do danych z tabel podrzędnych.
• Relacje pomiędzy tabelami w modelu sieciowym noszą nazwę struktury grupowej, w której jedna tabela jest właścicielem a inne tabele są członkami struktury.
• Struktury grupowe umożliwiają realizacje relacji jeden-dowielu pomiędzy tabelami.
• Jeśli zostanie zmieniona struktura bazy danych, zmiany wymaga również aplikacja.
• Trudna zmiana, modyfikacja struktury raz utworzonych bazach danych.

MODEL RELACYJNY (RDMBS)
W relacyjnych b.d. dane przechowujemy w dwuwymiarowych tabelach. (wiersze i kolumny). Każda z tabel składa się z rekordów oraz pól. Fizyczna kolejność pól i rekordów jest tutaj bez znaczenia. Każdy rekord jest wyróżniony przez unikatową wartość – klucz.

Najczęściej używanym obecnie modelem baz danych jest model relacyjny.
• Ojcem koncepcji relacyjnego modelu bazy danych jest dr E.F. Codd. z firmy IBM
• W tym modelu jest najłatwiej zaimplementować kontroli integralność danych po przez stosowaniu ograniczeniach (ang. constraints).
• Łatwo można zmieniać struktur bazy danych.
• Użytkownicy widzą logiczny sposób przechowywania danych i nie muszą znać ani rozumieć fizycznego sposobu ich przechowywania.
• Łatwiejszy proces odczytywania danych.

- *Obiektowa baza danych* (ang. *object-oriented database*) to taka baza, w której dane można przechowywać, definiować i korzystać z nich za pośrednictwem języków programowania obiektowego.
  - Języki programowania obiektowego - C++, Visual Basic, Java.
- W obiektowych bazach wyróżniamy dwie podstawowe struktury:
  - obiekty (ang. *objects*) strukturami posiadającymi identyfikatory umożliwiające tworzenie powiązań z innymi obiektami
  - literały (ang. *literals*) to wartości związane z obiektami, które nie posiadają identyfikatorów.

- Obiekty mogą dziedziczyć właściwości od innych obiektów.
- Poprzez nadawanie wartości właściwościom obiektów określa się między innymi sposób ich zachowania.
- W przypadku bazy obiektowej język programowania obiektowego służy zarówno do definiowania struktury bazy danych, jak i do tworzenia aplikacji będącej interfejsem tej bazy.
- Obiektowy model danych określa
  - zbiór obiektów;
  - ich stan;
  - uch zachowanie;
  - relacje między nimi

Rekord (ang.record) zwany także krotką lub wierszem, to pozioma struktura danych opisująca jeden obiekt. Rekord składa się z pól opisujących dokładnie cechy obiektu np. pojedynczego pracownika.

Pole (ang. field) zwane także atrybutem lub kolumną, to struktura danych opisująca pojedynczą daną w rekordzie np. nazwisko pracownika.

Tabela (ang. table) - nazywamy zbiór rekordów opisujących obiekty w sposób ujednolicony tj. każdy rekord posiada te same nazwy pól. Uwaga: w niektórych systemach baza danych np. dBase każda tabela nazywana jest bazą danych i jest przechowywana w oddzielnych plikach.

Klucz podstawowy (ang. primary key field, primary key) zwany
też kluczem głównym to jedno lub więcej pól, których wartość jednoznacznie identyfikuje każdy rekord w tabeli. Taka cecha klucza nazywana jest unikatowością. Klucz podstawowy służy do powiązania rekordów w jednej tabeli z rekordami z innej tabeli. Klucz podstawowy jest nazywany kluczem obcym, jeśli odwołuje się do innej tabeli. Na przykład, w bazie pracowników kluczem podstawowym może być numer ewidencyjny

pracownika.

Dana (ang.data) najmniejsza, elementarna jednostka informacji o obiekcie będąca przedmiotem przetwarzania komputerowego.

Relacyjna baza danych (ang.database) zbiór danych w postaci tabel połączonych relacjami.

Typ danej (ang. data type) - rodzaj danej, czyli forma zapisu informacji:

- **znakowy** (ang.character) dana może przybierać tylko wartości znaków pisarskich
- **liczbowy** (ang.number) dana może przechowywać tylko liczby
- **logiczny** (ang.logical) dana może przybierać tylko dwie wartości: prawda, fałsz (tak, nie)
- **data** (ang.date) dana może przyjmować postać daty i czasu np. rok.miesiąc.dzień godz:min:sek
- **alfanumeryczny** (ang.alphanumeric) dana może przybierać wartości znaków ASCII oraz cyfry
- **numeryczny** (ang.numeric) wartościami danej mogą być tylko cyfry i znaki: + (plus), - (minus).
- **walutowy** (ang.currency) dana może przyjmować wartości liczbowe razem z symbolem waluty
- **notatnikowy** (ang.memo) dana może być oddzielnym zbiorem tekstowym służącym do przechowywania dowolnych opisów.
- **binarny** (ang.binary) dana może być np. plikiem dźwiękowym lub filmowym.
- **graficzny** (ang.graphic) dana przechowuje grafikę np. rysunki.
- **obiektowy** (ang.OLE) dana przechowuje obiekty do których dostęp dokonuje się za pomocą techniki OLE (ang. object linking and embleding), czyli obiektów tworzonych przez inne aplikacje.

Format danej (ang. data format) postać wprowadzania i wyświetlania danej np. format 99-999 oznacza, że dana numeryczna może być wyświetlona jako ciąg 2 cyfr, pojedynczej kreski i 3 cyfr. Format ten może służyć do wprowadzani i wyświetlania kodu pocztowego.

Relacja (ang.relation)

Po podzieleniu danych na tabele i zdefiniowaniu pól kluczy podstawowych trzeba wprowadzić do systemu bazy danych informacje na temat sposobu poprawnego łączenia powiązanych danych w logiczną całość. W tym celu definiuje się relacje między tabelami.

Typy relacji (ang.relation types)

- **Relacja jeden-do-jednego**

W relacji jeden-do-jednego każdy rekord w tabeli A może mieć tylko jeden dopasowany rekord z tabeli B, i tak samo każdy rekord w tabeli B może mieć tylko jeden dopasowany rekord z tabeli A. Ten typ relacji spotyka się rzadko, ponieważ większość informacji powiązanych w ten sposób byłoby zawartych w jednej tabeli. Relacji jeden-do-jednego można używać do podziału tabeli z wieloma polami, do odizolowania części tabeli ze względów bezpieczeństwa, albo do przechowania informacji odnoszącej się tylko do podzbioru tabeli głównej.

- **Relacja jeden-do-wielu**

Relacja jeden-do-wielu jest najbardziej powszechnym typem relacji.

W relacji jeden-do-wielu rekord w tabeli A może mieć wiele dopasowanych do niego rekordów z tabeli B, ale rekord w tabeli B ma tylko jeden dopasowany rekord w tabeli A.

- **Relacja wiele-do-wielu**

W relacji wiele-do-wielu, rekord w tabeli A może mieć wiele dopasowanych do niego rekordów z tabeli B i tak samo rekord w tabeli B może mieć wiele dopasowanych do niego rekordów z tabeli A. Jest to możliwe tylko przez zdefiniowanie trzeciej tabeli (nazywanej tabelą łącza), której klucz podstawowy składa się z

dwóch pól ž kluczy obcych z tabel A i B. Relacja wiele-do-wielu jest w istocie dwiema relacjami jeden-do-wielu z trzecią tabelą. Na przykład, tabele "Zamówienia" i "Produkty" są powiązane relacją wiele-do-wielu zdefiniowaną przez utworzenie dwóch relacji jeden-do-wielu z tabelą "Opisy zamówień".

Sortowanie (ang.sorting)
Sortowaniem rekordów nazywamy ich porządkowanie według jakiegoś kryterium. Kryterium to nazwa lub nazwy pól według których odbywa się sortowanie. Przykładowo może być to sortowanie rosnące (sortuje wartości w porządku rosnącym (od A do Z, od 0 do 9) lub sortowanie malejące (sortuje wartości w porządku malejącym od Z do A, od 9 do 0).

Zapytanie (ang.query)
Zapytanie, czyli kwerenda to taka konstrukcja językowa, która pozwala na wyszukiwanie danych z bazy danych za pomocą zadawania pytań. Może to być specjalna konstrukcja języka programowania lub okno graficzne w którym należy podać parametry poszukiwanych danych. W zaawansowanych systemach baz danych kwerend można używać także do wyświetlania, zmiany i analizy danych.

Kwerenda wybierająca jest najczęściej używanym rodzajem kwerendy. Służy do otrzymywania danych z tabeli lub tabel i wyświetlania wyników w arkuszu danych, w którym można je następnie przeglądać. Kwerendy wybierające mogą być również używane do grupowania rekordów i obliczania sum, zliczeń, wyliczania średnich i przeprowadzania innych obliczeń.

Makro (ang. macro)
Ciąg akcji wykonywanych na tabelach, formularzach, raportach, kwerendach uruchamianych przyciskami umieszczonymi w oknie np. formularza lub gdy wystąpi jakieś zdarzenie np. kasowanie rekordu.

Filtr (ang. filter)
Filtr pozwala na wyszukiwanie rekordów spełniające pojedyncze kryterium lub wiele kryteriów albo sortować rekordy

w porządku rosnącym lub malejącym.

Formularz (ang. screen form)
Formularz, czyli tzw. formatka ekranowa służy do wygodnego wprowadzania, edytowania i usuwania danych w tabeli. Wymienione operacje wykonuje się za pomocą okna w którym użytkownik obsługuje pola. Znaczenie pól opisane jest za pomocą etykiet, czyli nazw pól.

Raport (ang.report)
Raportem nazywamy konstrukcję systemu bazy danych, która służy do definiowania postaci i zawartości danych pobieranych z tabel, a następnie umieszczanych na wydruku.

Procedura (ang.procedure)
Procedurą nazywamy serię poleceń zapisaną w języku programowania baz danych, służącą do wykonywania obsługi na elementach bazy: **tabelach, formularzach, raportach, kwerendach**.

Cechy bazy danych:

- **trwałość danych** - przechowywanie danych na pamięci masowej
- **niezależność danych** – pozwala na większą elastyczność
- **DBMS** – gwarantuje niezależność fizyczną, przejmuję na siebie zadania
- **ochrona danych**
- **integralność danych**

DBMS:

- **tworzenie struktur bazy danych**
- **operacja CRUD**(create, read, update, delete)
- **obsługa zapytań**
- **generowanie raportów**
- **administracja bazą**

Modele Baz Danych:

- **Jednorodny**
- **Hierarchiczny**
- **Sieciowy**
- **Relacyjny**
- **Obiektowy**

13 Postulatów „Codda”

- **System** musi być kwalifikowany jako relacyjny, jako baza danych i jako system zarządzania.
- **Postulat informacyjny** – dane są reprezentowane jedynie przez wartości atrybutów w wierszach tabel (w krotkach).
- **Postulat dostępu** – każda wartość w bazie danych jest dostępna poprzez podanie nazwy tabeli, atrybutu i wartości klucza podstawowego (głównego).
- **Postulat dotyczący wartości NULL** – dostępna jest specjalna wartość NULL dla reprezentacji zarówno wartości nieokreślonej, jak i nieadekwatnej, inna od wszystkich i podlegająca przetwarzaniu.
- **Postulat dotyczący katalogu** – wymaga się, aby system obsługiwał wbudowany katalog relacyjny z bieżącym dostępem dla uprawnionych użytkowników używających języka zapytań.
- **Postulat języka danych** – system musi dostarczać pełny język przetwarzania danych, który może być używany zarówno w trybie interaktywnym, jak i w obrębie programów, obsługuje operacje definiowania danych, operacje manipulowania danymi, ograniczenia związane z bezpieczeństwem i integralnością oraz operacje zarządzania transakcji.
- **Postulat modyfikowalności perspektyw** – system musi umożliwiać modyfikowanie perspektyw, o ile jest ono semantycznie realizowalne.

- **Postulat modyfikowalności danych** – system musi umożliwiać operacje modyfikacji danych, musi obsługiwać operacje INSERT, UPDATE oraz DELETE.
- **Postulat fizycznej niezależności danych** – zmiany fizycznej reprezentacji danych i organizacji dostępu nie wpływają na aplikacje.
- **Postulat logicznej niezależności danych** – zmiany wartości w tabelach nie wpływają na aplikacje.
- **Postulat niezależności więzów spójności** – więzy spójności są definiowane w bazie i nie zależą od aplikacji.
- **Postulat niezależności dystrybucyjnej** – działanie aplikacji nie zależy od modyfikacji i dystrybucji bazy.
- **Postulat bezpieczeństwa względem operacji niskiego poziomu** – operacje niskiego poziomu nie mogą naruszać modelu relacyjnego i więzów spójności.