Представяне на данните в информационни системи

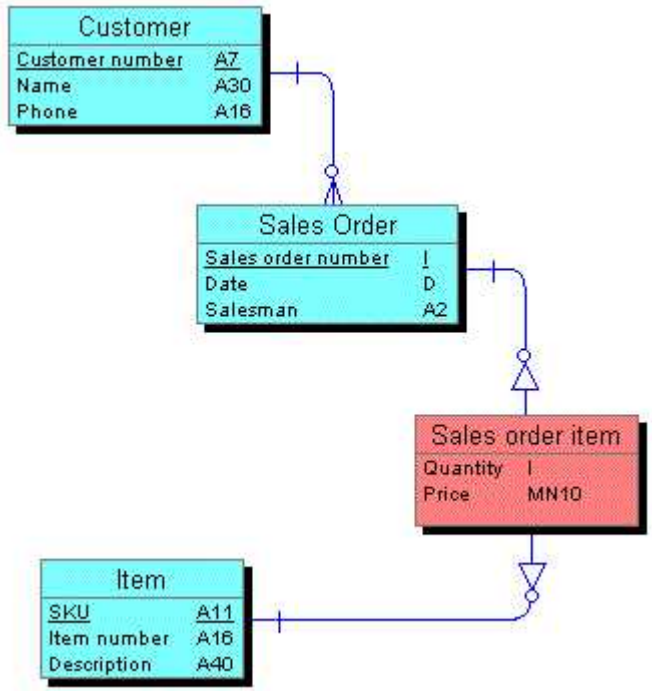
Представяне на данните в ИС

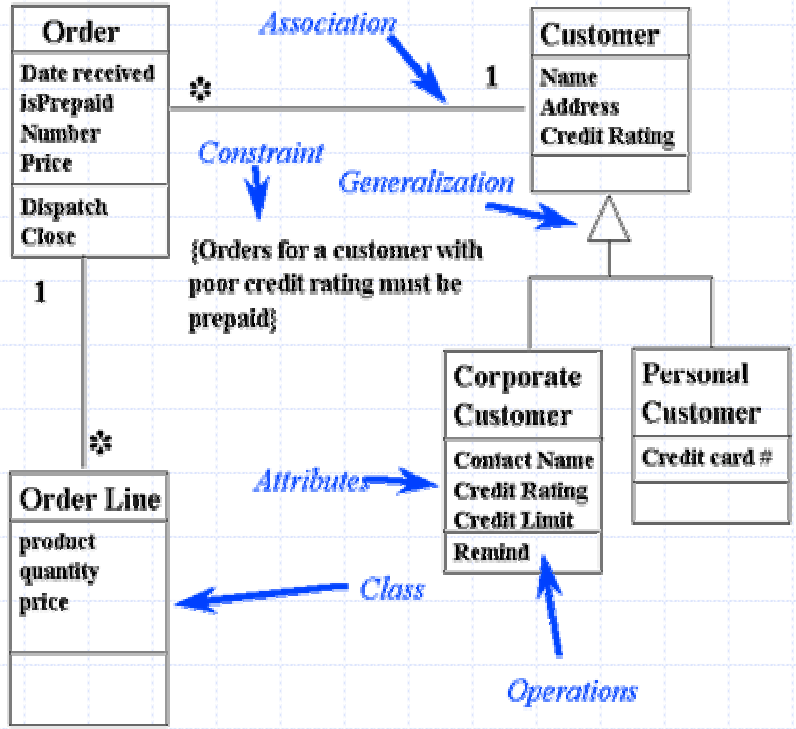
- Модели на данните
  - изграждане на концептуална схема на системата
  - разкриване на изискванията относно
    - необходимата извличана информация
    - операции по обработка на данни
  - алтернативни модели
- Структури от данни
- Преобразуване на моделите в структури
- Бази от данни

Модели на данните

- Структурно-ориентирани модели
  - обекти и отношения (ER)
    - обекти: атрибути, екземпляри и стойности
    - отношения: асоциации между обектите
      - тип – възможността да съществува отношение
      - всеки обект има определена роля в отношението
  - йерархия, наследяване на обекти (Enhanced ER)
- Обектно-ориентирани модели
  - дефинира се структура от данните на системата и операциите с тях
    - класове обекти - интерфейси
    - атрибути – свойства на обектите
    - отношения между обектите – свойство на класа, определящо ролята на този клас в отношението
  - клас диаграма

Модели на данните





Представяне на данните

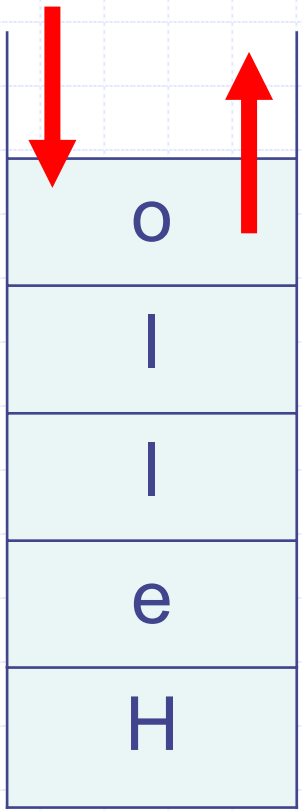
- Структуриране
  - представяне на данните в паметта
  - представяне на отношенията между данните по подходящ начин
- Операции с данните
  - Търсене
  - Добавяне
  - Изтриване

Видове структури

- Логически структури
  - прости / сложни
  - статични / динамични
  - външни / вътрешни

Структури

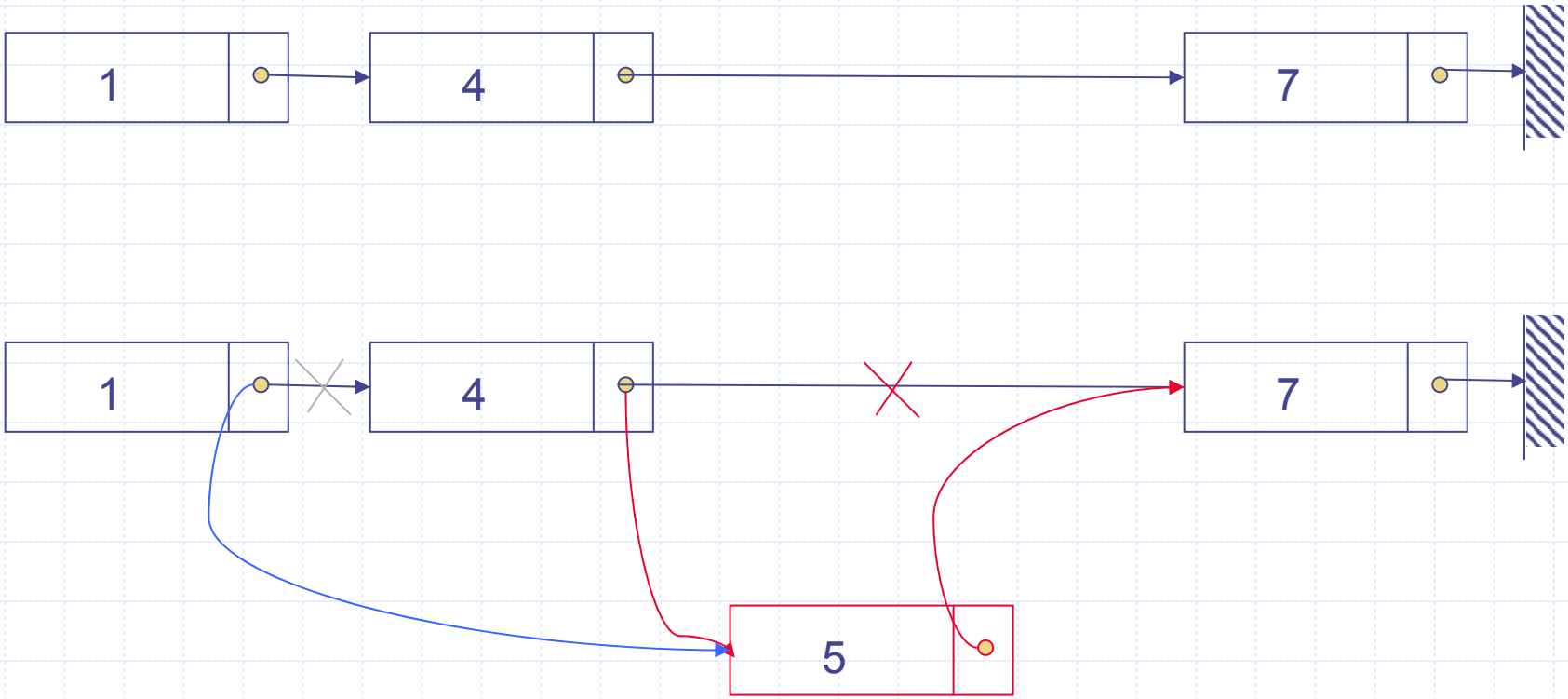
Стек



Опашка

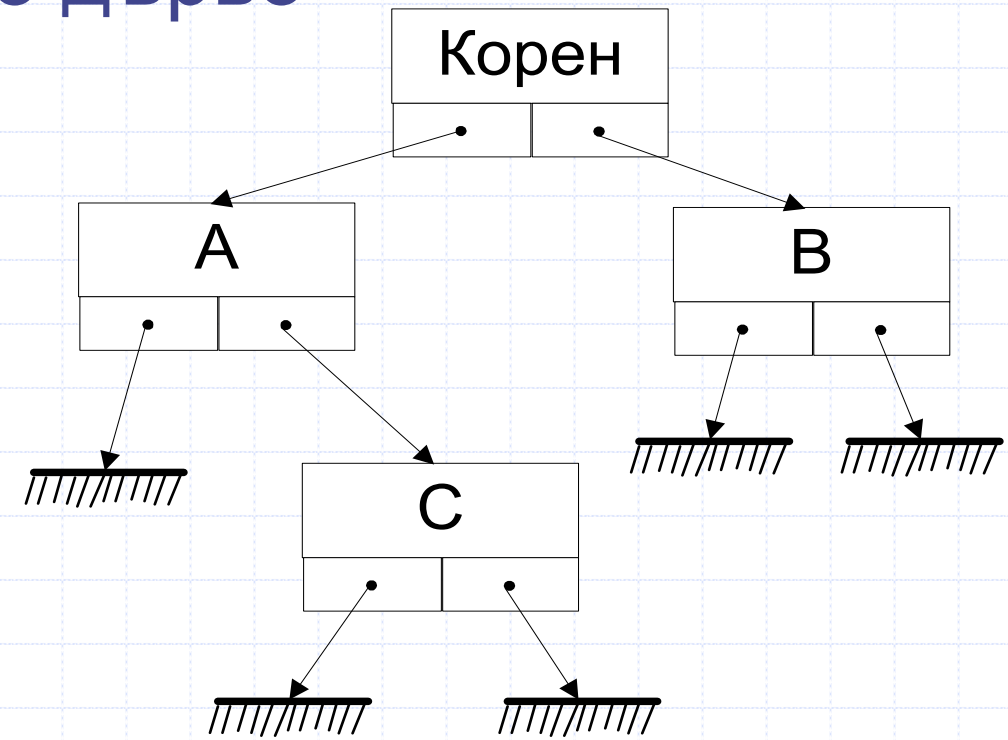
Структури

Линеен списък



Структури

Двоично дърво



Файлове с данни

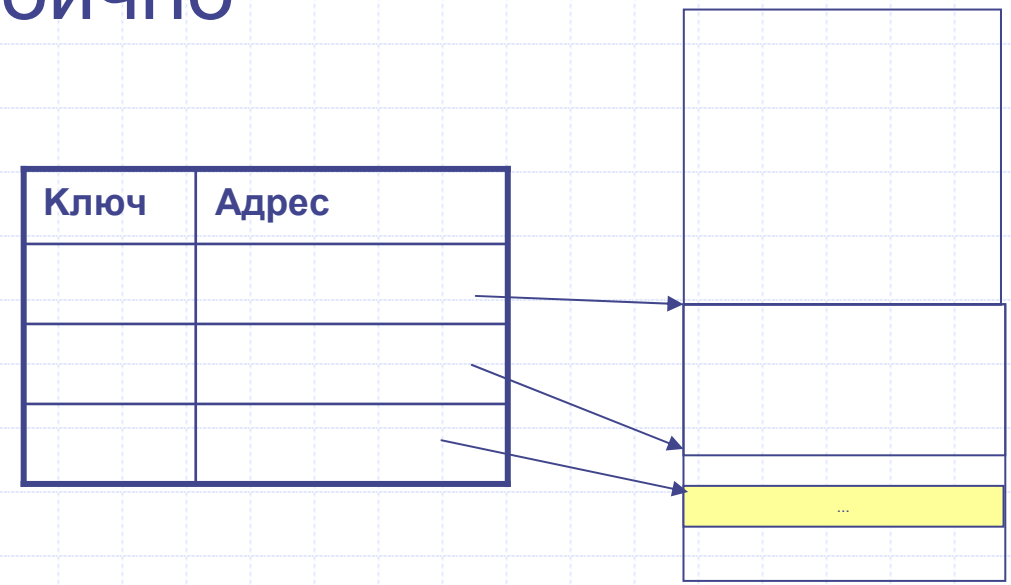
- Видове файлове
  - форматирани / неформатирани
  - текстови / бинарни
- Операции с данните във файл
  - търсене
  - сортиране
  - индексиране

Методи за търсене

- Последователно
  - несортиран файл
- Двоично
  - сортиран файл
- Блоково
  - сортиран файл
- Индексно
  - несортиран файл + индексен файл

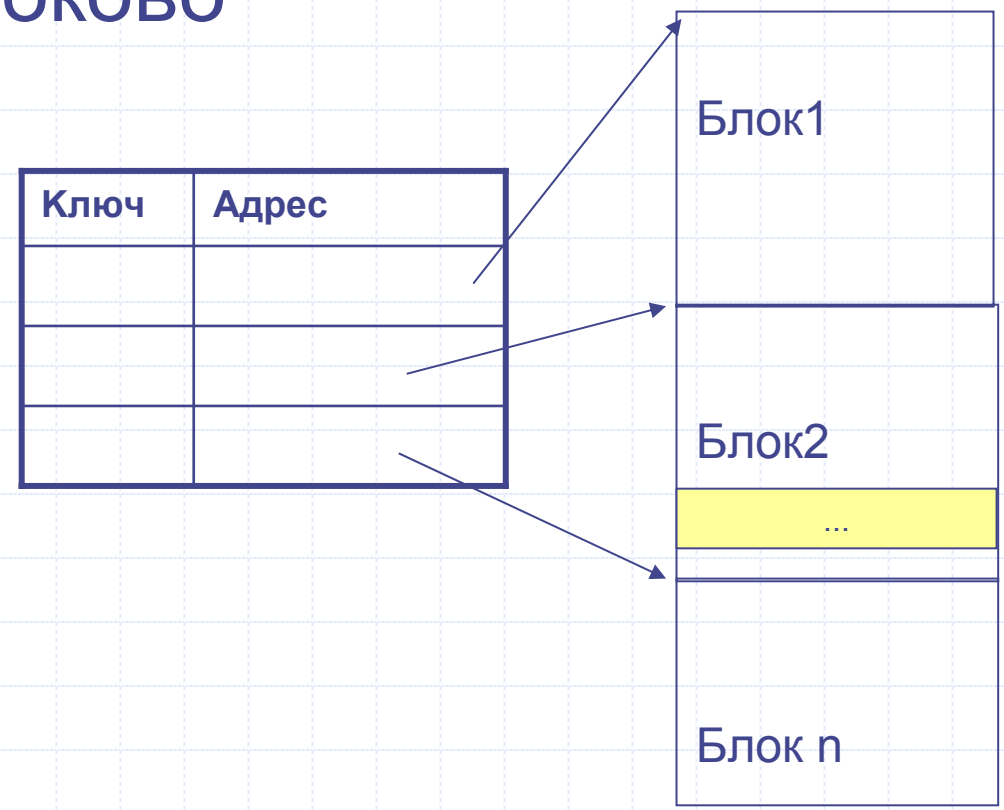
Индексно

- Двоично



Индексно

- Блоково



Преобразуване на моделите в структури

- Подходи за реализация
  - Файлова система

    програма+ файлове с данни => файл с резултат
  - Системи с база от данни

    файлове с данни + СУБД + програма => резултат

База от данни

- **Определения**
  - съвкупност от логически свързани данни, структурирани за ефективна обработка
  - БД е структурирано множество от данни, записано на носител, който е достъпен за четене и запис чрез компютър с цел да обслужи едновременно много потребители поотделно и за разумно време.
  - Множество от логически свързани и съхранявани заедно данни без излишък, които могат да бъдат използвани от различни приложения; начинът на съхранение е независим от приложенията и се използват общи процедури за промяна, добавяне и за търсене.
  - Множество от данни структурирани според една схема от данни.
  - Данни, които директно и едновременно са достъпни от различни, взаимно конкуриращи се потребители

Сравнение на подходите

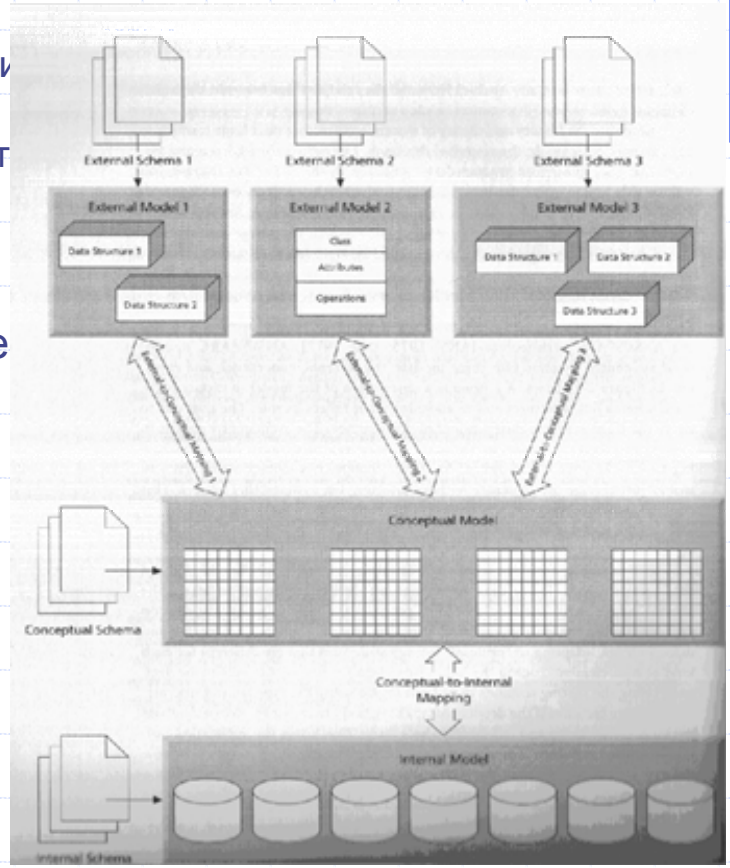
- Недостатъците на файлова организация:
  - дублиране на данни
  - несъвместимост
  - трудно поддържане
- Предимствата на база от данни:
  - намален излишък на данни;
  - администрирането е централизирано, което позволява да се съблюдават определени стандарти, даващи възможност за обмен на данни с други системи;
  - управление на правата на достъп, което дава по-голяма сигурност;
  - глобална нормализация на системите.

Архитектура на база от данни

- Архитектура
  - Описание на структурата на данните
  - Описание на поведението им
  - Правила за отношенията между тях
- Трисхемна архитектура
  - ANSI / SPARC, 1975
  - Системата се представя чрез 3 модела:
    - външен
    - концептуален
    - вътрешен
  - Схема: описание на модел
  - Схемите са независими помежду си
  - Описват се с различни средства
  - Резултат: програмна и апаратна независимост на данните

Три-схемна архитектура

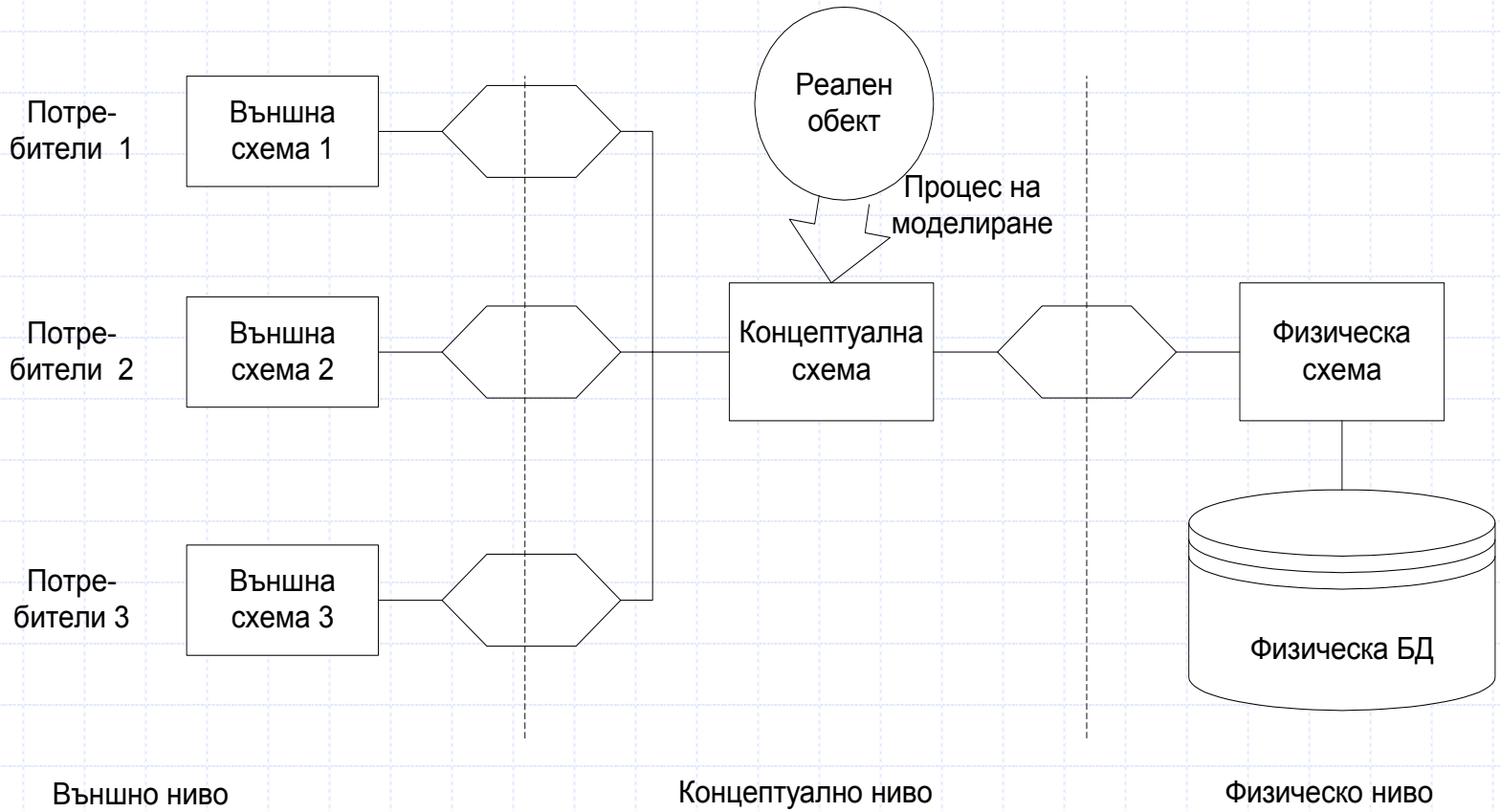
- външни - визуализация на данните за потребителите
  - Различни модели за различни потребители, създадени от различни приложения
  - Един външен модел съответства на целия или на част от логическия модел
  - Езици за управление на външни модели - програмни езици от високо ниво (C / C++, Java и др.)
- концептуални - логическа организация на данните
  - Единствен логически модел
  - Отразява структурата на данните
  - Осигурява логическа независимост на данните
  - Промяна на външен модел не изисква промяна на логическия модел
  - Езици за управление: **DMLs, SQL**
- вътрешни - физическа организация на запомнящата среда
  - Структури памет
  - Методи за достъп
  - Оптимизация и настройка
  - Управление на риска



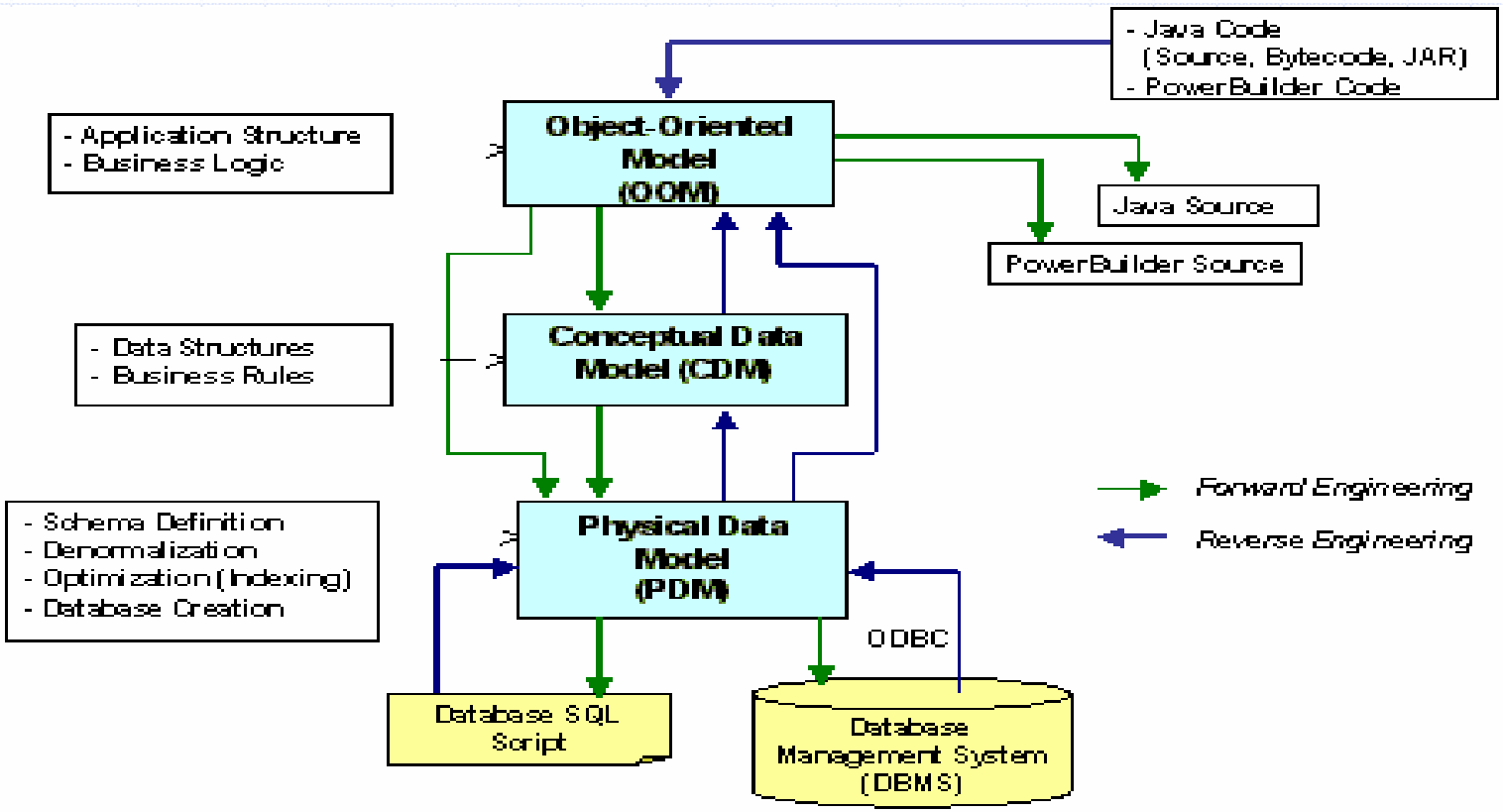
Технология на реализация

- релационни - RDBMS
  таблици, правила + SQL
- обектно-ориентирани – OODBMS
  класове и методи на класовете
- обектно-релационни - ORDBMS
  релационни структури + допълнителни типове обекти и методи, описващи тяхното поведение

Структурен подход

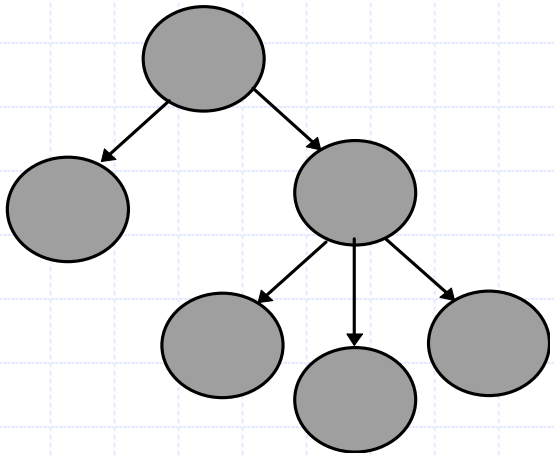


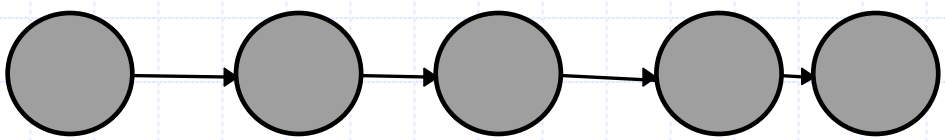
OO подход

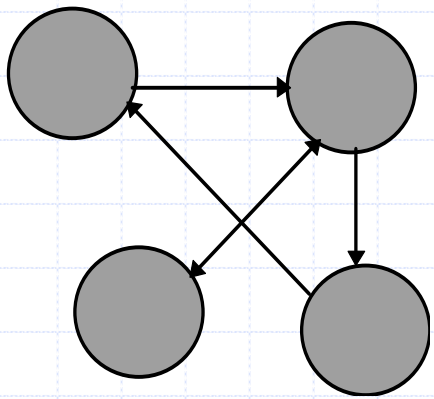

- Java Code
(Source, Bytecode, JAR)
- PowerBuilder Code
- Application Structure
- Business Logic
Object-Oriented Model (OOM)
Java Source
PowerBuilder Source
- Data Structures
- Business Rules
Conceptual Data Model (CDM)
- Schema Definition
- Denormalization
- Optimization (Indexing)
- Database Creation
Physical Data Model (PDM)
Forward Engineering
Reverse Engineering
ODBC
Database SQL Script
Database Management System (DBMS)

Видове концептуални модели

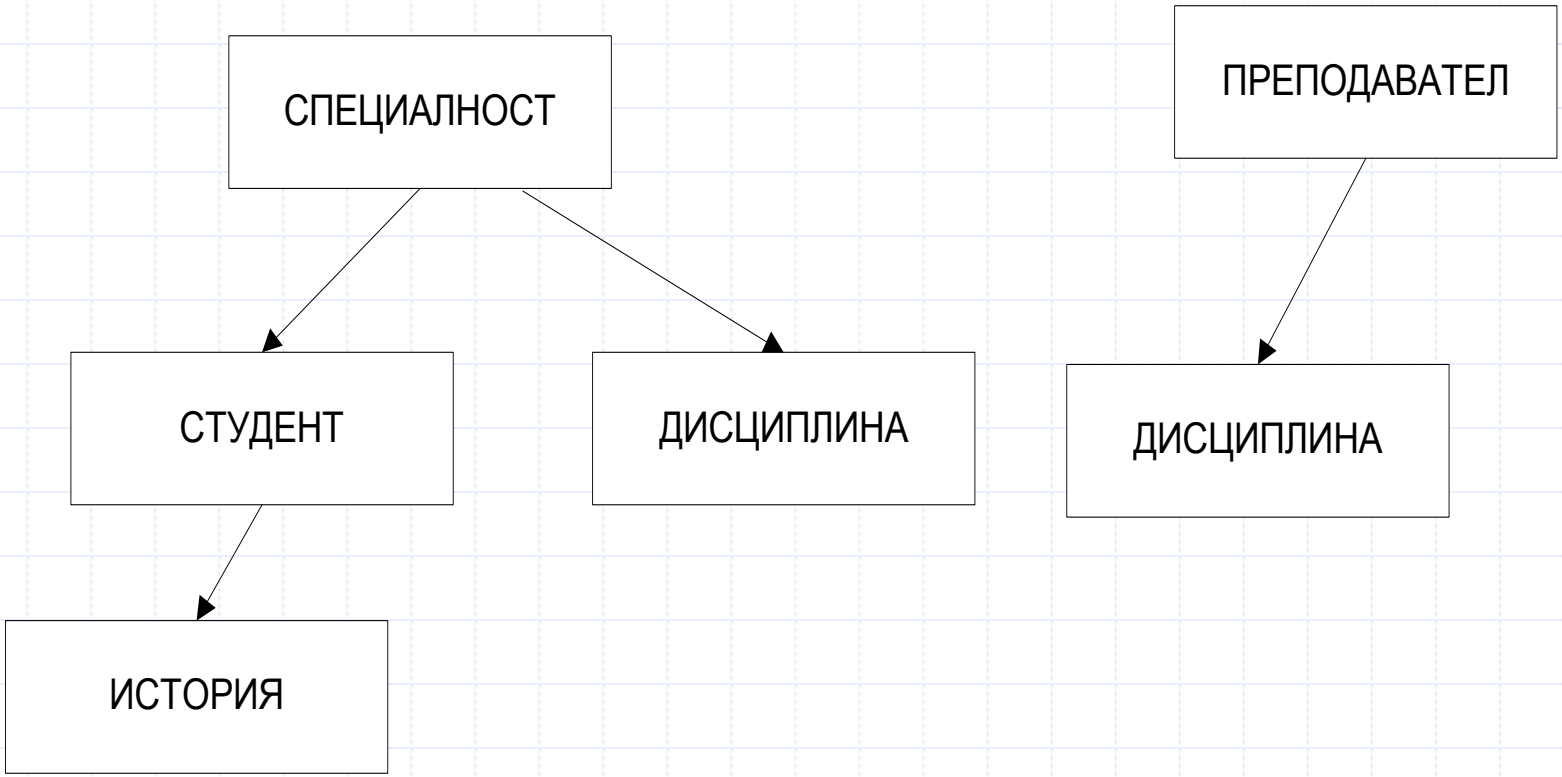
- Структура дърво – йерархичен модел
- Структура граф – мрежов модел
- Структура таблица – релационен модел



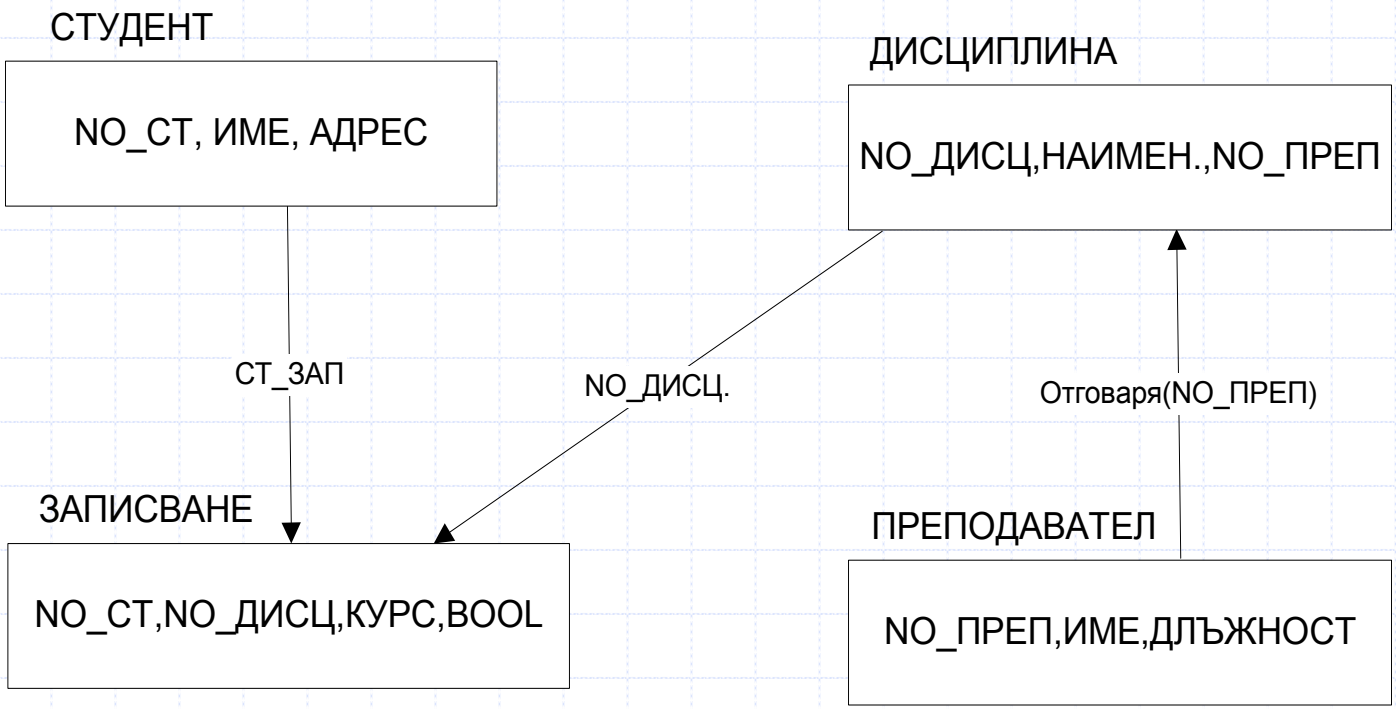




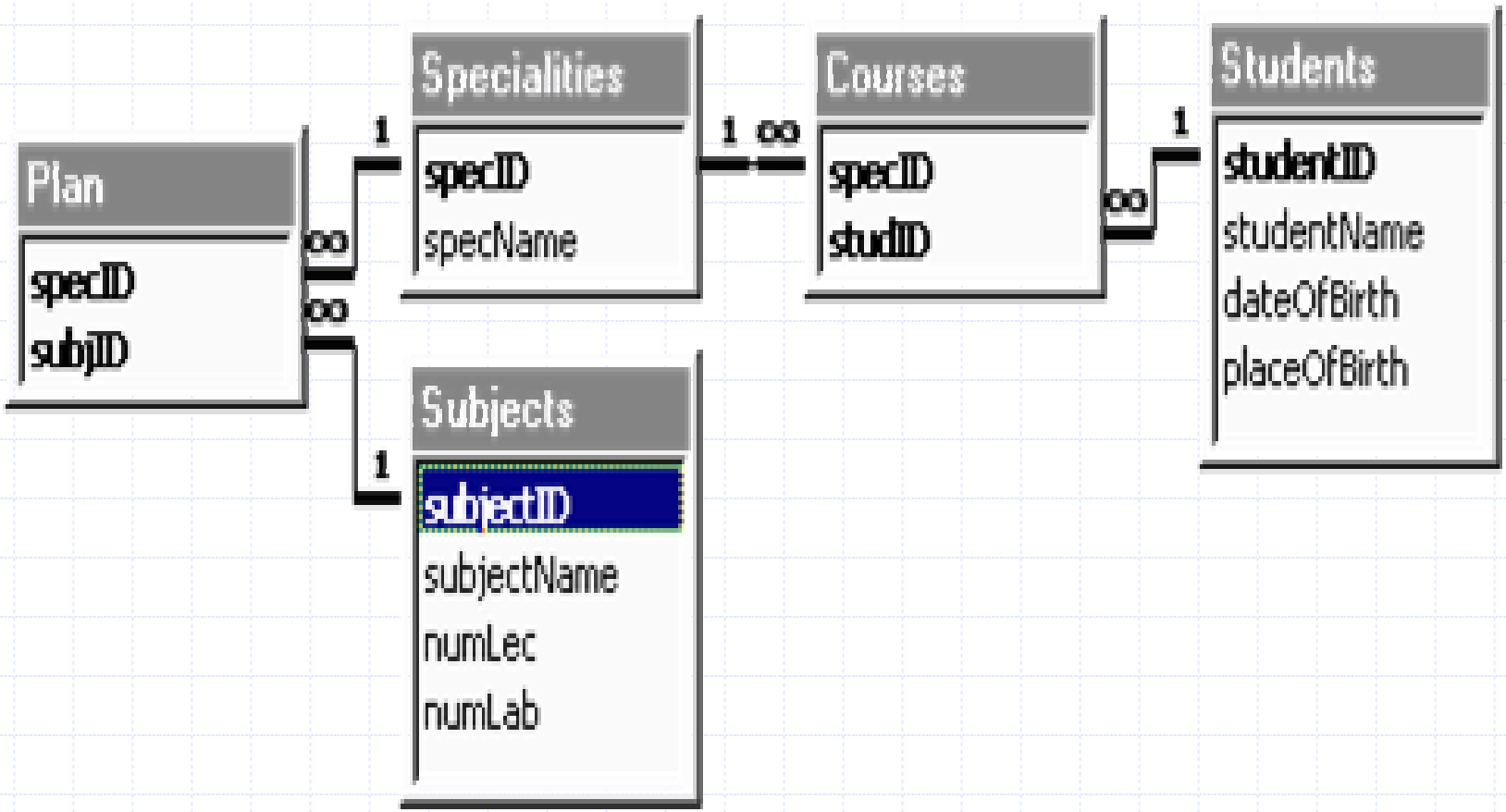
Йерархична схема



Мрежова схема



Релационна схема



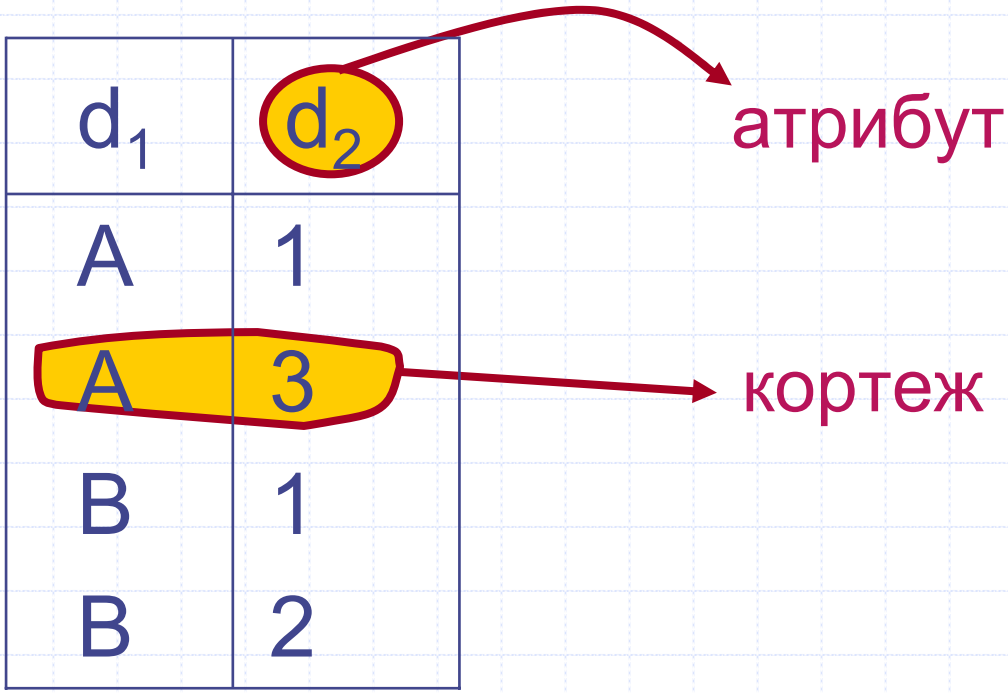
Релационни схеми

- Релационен модел на данните
- Основен обект
  - таблица
- Математически апарат
  - релационна алгебра
  - теория на множествата

Релационен модел на данните

Релация -> таблица

R_1 = { (A, 1), (A, 3), (B, 1), (B, 2) }



Релационна алгебра

Релация:

крайно под-множество R на декартовото произведение на множество домейни

$M = \{ D_1 \times D_2 \times D_3 \times \ldots, \times D_n \}$

Домейн:

множество от възможни стойности

$D_i = \{ V_1, V_2, V_3, \ldots, V_k \}$

Примери:

$D_1 = \{ A, B \}$ $D_2 = \{ 1, 2, 3 \}$

$M = D_1 \times D_2 \Rightarrow \{ (A, 1), (A, 2), (A, 3), (B, 1), (B, 2), (B, 3) \}$

$R_1 = \{ (A, 1), (A, 3), (B, 1), (B, 2) \}$

$R_2 = \{ (A, 1), (A, 2), (B, 3), (B, 2), (B, 3) \}$

Релационен модел

- **Релацията** може да се представи като двумерна таблица, където стълбовете отговарят на домените, редовете – на кортежите. Може да се види, че разместването на редове или стълбове не променя релацията
- **Атрибутът** е стълбът в една релация, характеризиран от име. Това прави подредбата на стълбовете без значение
- **Релационната схема** е името на релацията последвано от списък на атрибутите с техните домени (структурата на таблицата). Тази схема представлява *проект* (***intention***) на релацията.
- Таблицата с кортежите (съдържанието) е една ***реализация*** (***extension***).

Релационен модел

- Една реализация на релационната схема SR = (X, C) е релация, чиято схема е X и стойностите ѝ удовлетворяват всички ограничения на C
- Релационна база от данни: БД, чиято схема е множество от релационни схеми
- Тъй като релацията е една реализация на схемата, понятието релация съответства на състояние на БД, което може да се променя вследствие на операциите на обновяване:
  - Добавяне на кортеж в релацията.
  - Премахване на кортеж от релацията
  - Модификация на кортеж
  - Търсене

Таблица

- една таблица – една група обекти, клас
- един ред – един екземпляр от класа
- една колона – един атрибут на този клас обекти
- една клетка – една стойност на атрибута за екземпляра
- всеки екземпляр е уникален, т.е. няма повтарящи се редове
- неподредена (подредбата на редовете няма значение)

Пример

Таблица

Personal

Атрибути

EGN

ime

address

Екземпляри

4

Personal (EGN, ime, address)

EGN	ime	address
1234	Боян	Слатина
1244	Антон	Младост
1108	Светла	Люлин
8543	Мила	Дианабад

Ключ на таблицата

- Идентификатор на екземпляра
- Съществува поне един
- Има уникална стойност за всеки екземпляр
- Осигурява уникалност на реда в таблицата
- Нарича се първичен ключ (*Primary Key, PK*)
- Една релация може да има повече от един ключ (кандидат ключове или над-ключове), но само един от тях се избира за първичен ключ.

Първичен ключ

- Подмножеството от атрибути X е **ключ** на релацията R(A1,A2,…,An), ако X $\rightarrow$ A1 A2 …An и не съществува такова подмножество Y $\subseteq$X, че Y $\rightarrow$ A1 A2 …An
- Множество от атрибути X на класа, за които са изпълнени условията:

1. Няма два екземпляра, за които е (теоретично) възможно да имат еднакви стойности на тези атрибути.

2. Няма под-множество на X, за което да е валидно условието 1.

Пример

Personal (EGN, ime, address)

EGN – първичен ключ

EGN	ime	address
1234	Боян	Слатина
6543	Еми	Младост
1244	Антон	Младост
1108	Светла	Люлин
7856	Боян	Обеля
8543	Мила	Дианабад

Съставен първичен ключ

- Когато няма един подходящ атрибут, се търси множество от повече атрибути, които заедно идентифицират екземплярите еднозначно

EGN	date	workhours
1236	16/02/04	8
1236	17/02/04	4
1238	16/02/04	8

Work(EGN,date, workhours)

ID - идентификатор

- Когато няма един подходящ атрибут, алтернативно:

  добавя се атрибут – идентификатор на екземпляра

Телефонен указател

фамилия	име	адрес	телефон
Крайников	Йосиф А.	Х. Димитър 66	454529
Крайниковски	Асен И.	Ивац Войвода 20	449949
Крайнин	Христо К.	Червена роза 31	668819
Крайничанец	Драган Д.	В. Левски 52	539282

Ключ

ID	фамилия	име	адрес	телефон
1	Крайников	Йосиф А.	Х. Димитър 66	454529
2	Крайниковски	Асен И.	Ивац Войвода 20	449949
3	Крайнин	Христо К.	Червена роза 31	668819
4	Крайничанец	Драган Д.	В. Левски 52	539282

ID - ключ

Функционална зависимост

Нека е дадена релацията R(X, Y, Z) (Z е множество от атрибути, което евентуално е празно).
Казваме, че съществува функционална зависимост между X и Y означена с

$X \xrightarrow{R} Y$ или $X \rightarrow Y$

ако и само ако, каквито и да са стойностите (a, b, c) и (a', b', c') върху X, Y и Z

$$\left|R(a,b,c)\right| \wedge \left|R(a,b',c')\right| \Rightarrow (b=b')$$

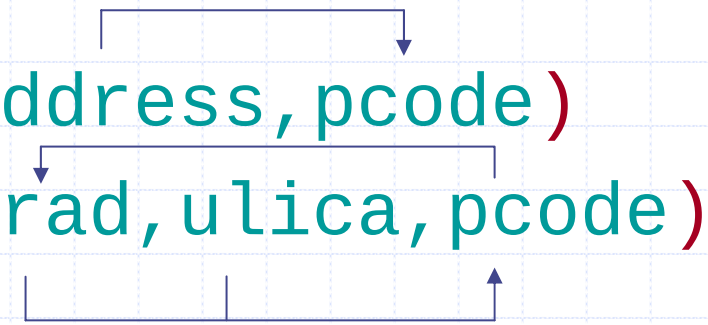
Функционална зависимост

Ако d_1 и d_2 са атрибути, то

$$\mathbf{d_1 \rightarrow d_2}$$

атрибут $\mathbf{d_2}$ е функционално зависим от атрибут $\mathbf{d_1}$, ако стойността на $\mathbf{d_1}$ еднозначно определя стойността $\mathbf{d_2}$.

Примери:



Ако два реда имат еднакви стойности за $\mathbf{d_1}$,

то те ще имат еднакви стойности за $\mathbf{d_2}$, също.

Функционална зависимост

- Функция на зависимостта
  - определя стойност на d_2 като функция на d_1
  - не е формула
  - определя се като за стойност a от атрибут d_1
    1. се намира ред в таблицата, за който стойността на d_1 е a
    2. се връща стойността на d_2 за този ред

Функционална зависимост

- Произлизащи от данните
- Следствия от други функционални зависимости

Аксиоми (Армстронг, 1974)

рефлексивност : ако $Y \subseteq X$, то $X \rightarrow Y$

добавяне : ако $X \rightarrow Y$ и W е множество от атрибути, то $XW \rightarrow YW$

транзитивност : ако $X \rightarrow Y$ и $Y \rightarrow Z$ то $X \rightarrow Z$

псевдо-транзитивност : ако $X \rightarrow Y$ и $YW \rightarrow Z$, то $XW \rightarrow Z$

обединение : ако $X \rightarrow Y$ и $X \rightarrow Z$, то $X \rightarrow YZ$

декомпозиция : ако $X \rightarrow YZ$, то $X \rightarrow Y$ и $X \rightarrow Z$

Интегритет

- Осигуряване на съответствие между данните в таблиците
- Постига се посредством
  - ключове
  - нормализация
  - отношения между таблиците

Нормализация

- Процес на осигуряване на оптимална структура на базата от данни
- Декомпозиция на релация R(A1,..., An) и заместването ù с множеството релации R1,…, Rn, получени чрез проекции така, че
  - R и R1 * R2 * **...** * Rn имат една и съща схема.
- Трансформиране на структурата на таблиците, с цел оптимизиране на функционалните зависимости на атрибутите им
- Основава се на взаимозависимостта на данните
- Води до
  - икономия на памет, посредством премахване на излишеството - отстраняване на повтарящи се данни
  - бързодействие, посредством ускоряване на търсенето и намирането на данни

Нормални форми

1NF: всеки атрибут има по една стойност за екземпляр (атомарни данни)

2NF: всеки атрибут е функционално зависим от целия първичен ключ, но не от негови отделни елементи

3NF: един атрибут не може да бъде функционално зависим от друг (неключов) атрибут

4NF: множествени функционални зависимости

5NF: относно зависимости между свързани обекти

1NF

EMPLOYEE (No, name, Child (firstname, age))

No	NAME	CHILD	
		FIRSTNAME	AGE
500	DUPONT	ANDRE	10
501	DURAND	JEAN	11
501	DURAND	PIERRE	12
510	LEFEBVRE	PAUL	13
510	LEFEBVRE	JACQUES	14

EMPLOYEE	No	NAME
	500	DUPONT
	501	DURAND
	510	LEFEBVRE

CHILDREN	No	FirstName	age
	500	André	10
	501	Jean	11
	501	Pierre	12
	510	Paul	13
	510	Jacques	14

1NF

MOVIE(No, Name, Director, Actor1, Actor2, Actor3)

MOVIE	No	Name	Director	Actor1	Actor2	Actor3
	1	The Silence of the Lambs	Jonathan Demme	Jodie Foster	Anthony Hopkins	
	2	The Sixth Sense	M. Night Shyamalan	Bruce Willis		

MOVIE	No	Name	Director
	1	The Silence of the Lambs	Jonathan Demme
	2	The Sixth Sense	M. Night Shyamalan

CAST	No	Actor
	1	Jodie Foster
	1	Anthony Hopkins
	2	Bruce Willis

2NF

Определение: Ще казваме, че Y напълно зависи от X, ако $X \rightarrow Y$ и няма подмножество $Z \subset X$, за което $Z \rightarrow Y$, т.е. Y зависи от целия X, но не зависи от никоя негова част

Една релация R е във 2NF ако и само ако:

1. Тя е в 1 NF
2. Всички неключови атрибути са напълно зависими от ключа на R.

Пример:

STOCK (part, warehouse, quant, address)

не е във 2NF

Декомпозиция

STOCK (part, warehouse, quant)

LOCAL(warehouse, address)

3NF

Една релация е в 3NF, ако и само ако:

1. Тя е във 2NF
2. Всички атрибути не принадлежащи на един ключ не зависят от неключов атрибут

Пример:

PERSONAL(employee, name, FirstName, Service, Address)

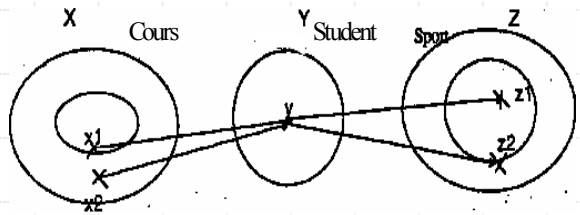
Address зависи само от **Service**.

Декомпозира се на **:**

PERSONAL(employee, name, FirstName, Service)
LOCAL(Service, Address)

4NF

STUDENTS	N	COURS	SPORT
	100	BD	Tennis
	100	BD	Football
	200	BD	Swimming
	200	AN	Swimming



X—» Z : x1yz1 e x2yz2∈ R ⇒ x1yz2 et x2yz1 ∈ R

Една релация е в 4NF ако и само ако единствените елементарни ФЗ са тези, при които един ключ определя един атрибут,

STUDENT (num, cours, sport) не е в 4NF

num -—» cours – R1(num, cours)

num —» sport– R2(num, sport).

НЕ-Нормални форми

1NF:

EGN	name	phone
1236	Симеон	~~9651833, 08881122~~

EGN	date	workhours	name	salary
1236	16/02/04	8	Симеон	160

2NF: Work(EGN,date, workhours, name, salary)

3NF: Work(EGN,date, workhours, name, salary)

Отношения

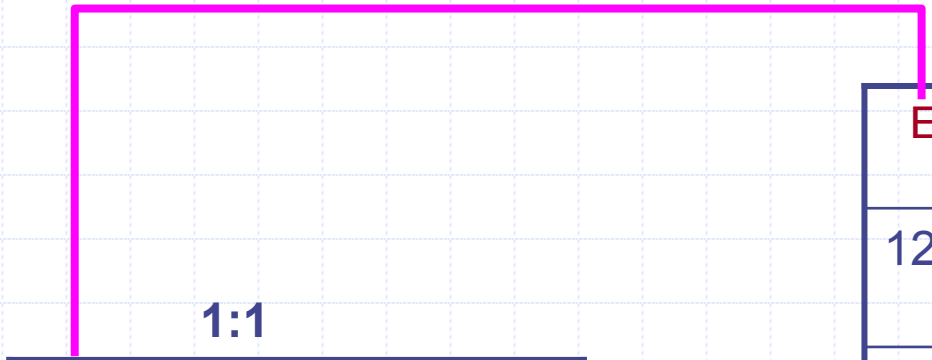
- установяване на зависимостите между класовете (таблиците)
- едно отношение (*Relationship*) е отношение между два класа
- кардиналност на отношението
  - свойство, което показва колко екземпляра от единия клас имат пряка връзка с екземпляри от другия клас

1:1 1:N N:1 N:M

Отношения

- Представят се чрез таблици, атрибути или и двете
  - атрибут външен ключ (1:N, N:1, 1:1)
  - таблица, чиито атрибути са ключове на асоциирани таблици (N:M)
  - сливащи се таблици (1:1)
- Външен ключ: **Множество от атрибути на релация, които се свързват с ключа на друга релация**
- Referential Integrity**: Ограничение на външния ключ: всяка стойност на атрибута външен ключ се свързва с обект от външната таблица**

Едно - към - едно

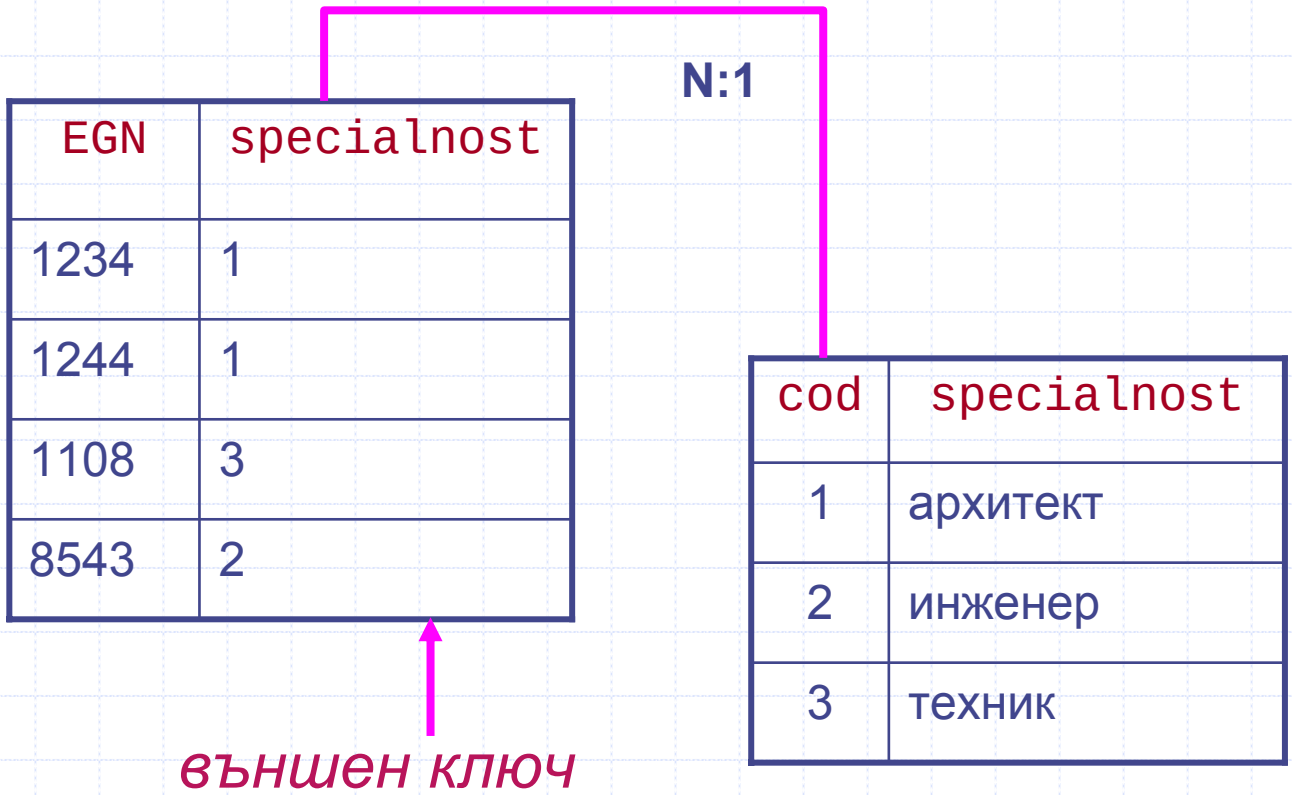


1:1

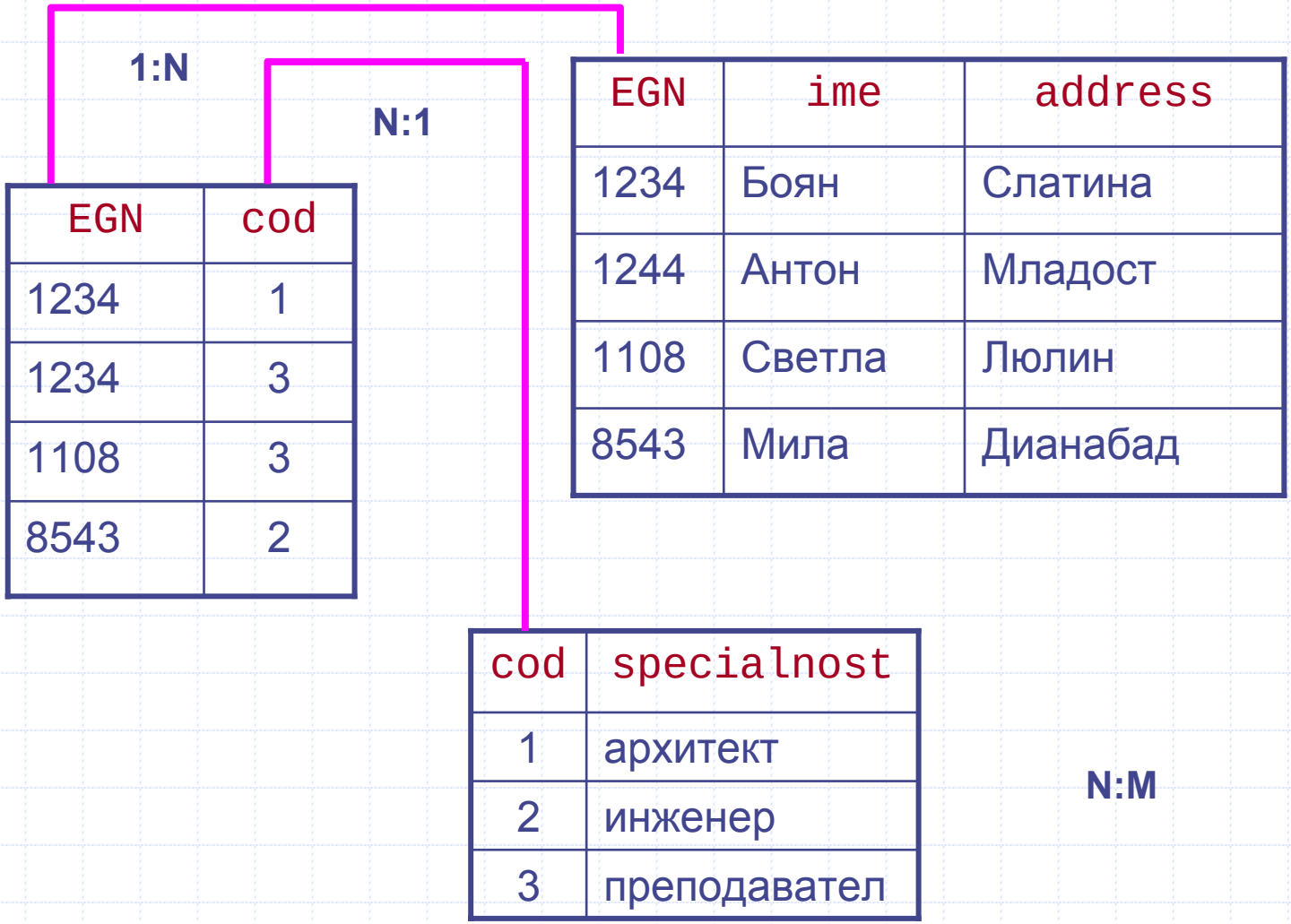
EGN	specialnost
1234	архитект
1244	архитект
1108	техник
8543	инженер

EGN	ime	address
1234	Боян	Слатина
1244	Антон	Младост
1108	Светла	Люлин
8543	Мила	Дианабад

Едно - към - много



Много - към - много



Обобщение

- Релационен модел
  - обекти (класове) – колекция от подобни екземпляри
  - атрибути – свойство на обектите
  - стойности – стойност на атрибут за конкретен екземпляр
  - отношения – асоциация между два или повече обекти
  - кардиналност – свойство на отношение, ограничаващо броя на участията на един екземпляр в отношението
  - силен клас – който има ключ за идентификация на екземплярите - пример Employee
  - слаб клас – който няма собствен ключ, а за идентификация на екземплярите използва дискриминатор, partial key (съставен ключ: част от ключа произлиза от идентифициращо отношение със силен клас, друга част осигурява идентификацията на екземплярите на този клас) – пример date за PayStatement

Обобщение

- Предимства на релационния модел
  - Простота за потребителя
  - Независимост на потребителя от логическата и физическата структура, както и от методите за достъп до данните.
    - Понятието файл съществува само за администратора и е грижа на СУБД
  - Мощност и еднотипност на представянето
    - математическата теория позволява точно и алгоритмично проектиране на схемата
  - Сигурност на данните
    - контрол според съдържанието, структурата и контекста
  - Съществуване на непроцедурен интерфейс за неспециалисти.
  - Бурно развитие на комерсиални СУБД и създаването на 4-то поколение SQL и QBE.
- Недостатъци на релационния модел
  - Известна загуба на логическа независимост при нормализацията.
  - Необходимост от мощна СУБД

Релационни схеми

- Правила за трансформиране на релационен модел в релационна схема
  - силни класове
  - съставни атрибути
  - отношения
  - слаби класове
  - специализация и генерализация –наследяване
  - категоризация – използване, отношение между клас и обединение на класове
  - множествени атрибути

Правила

- Силни класове
  - Правило 1: За всеки клас се създава по една релация със същото име
  - Правило 2: За всеки атрибут на класа се създава един атрибут в релацията
  - Правило 3: Избира се *първичен ключ* на релацията
- Съставни атрибути
  - Правило 4: За всеки съставен атрибут на класа се създават прости атрибути на релацията (напр. addres)

Правила

- Отношения
  - Правило 5: За всяко отношение 1:N между класове S и T се създават допълнителни атрибути в T, като *външен ключ*, които се вземат от ключа на S
  - Правило 6: Към T се добавят и другите атрибути на отношението
  - Правило 7: За всяко отношение 1:1 между класове S и T в единия клас се създава като външен ключ ключът на другия клас
  - Правило 8: За всяко отношение M:N между класове S и T се създава нова релация, в която атрибути са ключовете на S и T, като външни ключове. Първичният ключ на тази релация е комбинация на тези атрибути. В релацията са добавят и други атрибути на отношението.

Правила

- Слаби класове

  Правило 9: За всеки клас се създава по една релация, в която има собствени атрибути и атрибути от всеки силен клас, свързан в отношение с този клас. Тези атрибути са първични ключове в силните класове и външни ключове тук. Първичният ключ тук е комбинация от външните и дискриминатора *partial key* (пример PayStatement за Employee)

Правила

- Специализация и генерализация (алтернативи)
  - Правило 10: За супер-класа C се създава релация. Добавя се дефиниращ атрибут за специализациите на C. За всеки суб-клас се създава нова релация със собствени атрибути и с атрибут ключ към C.
  - Правило 11: Една обща релация: За супер-класа C се създава релация. Добавя се дефиниращ атрибут за специализациите на C. За всеки суб-клас, към C се добавят атрибутите на този клас.
  - Правило 12: За всеки суб-клас се създава нова релация със собствени атрибути. В новите релации се добавят всички атрибути на супер-класа C . Ключът на релациите е комбинация от атрибутите на супер-класа.
- Категоризация
  - Правило 13: Всяко категоризиращо отношение C, свързано с P и супер-клас B, се замества с две отношения на P със суб-класовете на B като отношението между P и C.

Правила

- Множествени атрибути

  Правило 14: За всеки атрибут с множество стойности се създава нова релация, като на слаб клас (пример OtherPhones).