Înainte de a începe o discuție detaliată cu privire la modul de proiectare a bazelor de date, trebuie remarcat faptul că orice bază de date este o parte integrantă a unui sistem de informații (IS), ceea ce implică nu numai stocarea datelor, ci și prelucrarea acestora. Prin urmare, proiectarea datelor însoțește întotdeauna (și mai des precede) proiectarea algoritmilor pentru utilizarea lor. Aici vom lua în considerare toate etapele proiectării unui sistem informațional: de la modelarea funcțională a domeniului, până la construirea structurii unei baze de date relaționale.

**5.1 Pași de proiectare a datelor**

Să ne reamintim încă o dată definiția conceptului de "subiect":

**Un domeniu** este o parte a lumii reale care trebuie studiată pentru a organiza managementul și, în cele din urmă, automatizarea. Domeniul subiect este reprezentat de multe *fragmente* , de exemplu, o întreprindere - de magazine, management, contabilitate etc. Fiecare piesă a domeniului este caracterizată de o multitudine de *obiecte* și *procese care* utilizează obiecte, precum și de o multitudine de *utilizatori* , caracterizați prin vizualizări diferite asupra domeniului.

În teoria proiectării sistemelor informaționale, aria subiectului (sau, dacă doriți, întreaga lume reală în ansamblu) este de obicei considerată sub forma a trei reprezentări:

1. reprezentarea ariei subiectului așa cum există cu adevărat
2. modul în care o persoană o percepe (adică proiectantul bazei de date)
3. cum poate fi descris folosind simboluri.

Acestea. ei spun că avem de-a face cu realitatea, o descriere (reprezentare) a realității și cu date care reflectă această reprezentare.

Datele utilizate pentru a descrie aria subiectului sunt prezentate sub forma unei scheme pe trei niveluri (așa-numitul model ANSI / SPARC):



Reprezentarea externă (schema externă) a datelor este o colecție de cerințe de date de la o anumită funcție îndeplinită de utilizator. Diagrama conceptuală este o colecție completă a tuturor cerințelor de date, derivate din percepțiile utilizatorilor despre lumea reală. Schema internă este baza de date în sine.

De aici urmează etapele principale în care se împarte procesul de proiectare a unei baze de date a sistemului de informații:

1. **Proiectare conceptuală** - colectarea, analizarea și editarea cerințelor de date. Pentru aceasta, se desfășoară următoarele activități:
	* cercetarea domeniului subiect, studiul structurii sale informaționale
	* identificarea tuturor fragmentelor, fiecare dintre acestea fiind caracterizată de o vizualizare a utilizatorului, obiecte de informații și conexiuni între ele, procese peste obiecte de informații
	* modelarea și integrarea tuturor vederilor

La sfârșitul acestei etape, obținem un model conceptual care este invariant la structura bazei de date. Este adesea reprezentat ca un model entitate-relație.

1. **Proiectarea logică** este transformarea cerințelor de date în structuri de date. Ca rezultat, obținem o structură de bază de date orientată către SGBD și specificațiile programelor de aplicații. În această etapă, bazele de date sunt adesea modelate în raport cu diferite SGBD și se efectuează analiza comparativă a modelelor.
2. **Proiectare fizică** - definirea caracteristicilor de stocare a datelor, metode de acces etc.

Diferența dintre nivelurile de prezentare a datelor la fiecare etapă de proiectare este prezentată în tabelul următor:

|  |  |
| --- | --- |
| **NIVEL CONCEPTUAL*** entități
* atribute
* conexiuni
 | Vizualizarea analistului |
| **NIVEL LOGIC*** înregistrări
* elemente de date
* legături între înregistrări
 | Viziunea programatorului |
| **NIVEL FIZIC*** gruparea datelor
* indicii
* metode de acces
 | Vizualizare administrator |

Literatură:

* Tiori T., Fry J. Proiectarea structurilor de baze de date. M, 1985.
* Hubbard J. Proiectarea computerizată a bazelor de date. M, 1984.
* Boyko V.V., Savinkov V.M. Proiectarea bazelor de date ale sistemelor informatice. M, 1989.

Capitolul următor: [5.2 Instrumente de proiectare IC.](https://translate.googleusercontent.com/translate_c?depth=1&hl=ro&pto=aue&rurl=translate.google.com&sl=ru&sp=nmt4&tl=ro&u=http://www.mstu.edu.ru/study/materials/zelenkov/ch_5_2.html&usg=ALkJrhj4f4oGIWWsk2QY3MN8nqAaBACA8g)

Перед началом детального обсуждения способов проектирования баз данных необходимо отметить, что любая база данных является составной частью некой информационной системы (ИС), которая подразумевает не только хранение данных, но и их обработку. Поэтому, проектированию данных всегда сопутствует (а чаще предшествует) проектирование алгоритмов их использования. Здесь мы рассмотрим все этапы проектирования информационной системы: от функционального моделирования предметной области, до построения структуры реляционной базы данных.

**5.1.Этапы проектирования данных**

Напомним еще раз определение понятия "предметная область":

**Предметная область** - часть реального мира, подлежащая изучению с целью организации управления и, в конечном счете, автоматизации. Предметная область представляется множеством *фрагментов*, например, предприятие - цехами, дирекцией, бухгалтерией и т.д. Каждый фрагмент предметной области харакетризуется множеством *объектов* и *процессов*, использующих объекты, а также множеством *пользователей*, харакетризуемых различными взглядами на предметную область.

В теории проектирования информационных систем предметную область (или, если угодно, весь реальный мир в целом) принято рассматривать в виде трех представлений:

1. представление предметной области в том виде, как она реально существует
2. как ее воспринимает человек (имеется в виду проектировщик базы данных)
3. как она может быть описана с помощью символов.

Т.е. говорят, что мы имеем дело с реальностью, описанием (представлением) реальности и с данными, которые отражают это представление.

Данные, используемые для описания предметной области, представляются в виде трехуровневой схемы (так называемая модель ANSI/SPARC):



Внешнее представление (внешняя схема) данных является совокупностью требований к данным со стороны некоторой конкретной функции, выполняемой пользователем. Концептуальная схема является полной совокупностью всех требований к данным, полученной из пользовательских представлений о реальном мире. Внутренняя схема - это сама база данных.

Отсюда вытекают основные этапы, на которые разбивается процесс проектирования базы данных информационной системы:

1. **Концептуальное проектирование** - сбор, анализ и редактирование требований к данным. Для этого осуществляются следующие мероприятия:
	* обследование предметной области, изучение ее информационной структуры
	* выявление всех фрагментов, каждый из которых харакетризуется пользовательским представлением, информационными объектами и связями между ними, процессами над информационными объектами
	* моделирование и интеграция всех представлений

По окончании данного этапа получаем концептуальную модель, инвариантную к структуре базы данных. Часто она представляется в виде модели "сущность-связь".

1. **Логическое проектирование** - преобразование требований к данным в структуры данных. На выходе получаем СУБД-ориентированную структуру базы данных и спецификации прикладных программ. На этом этапе часто моделируют базы данных применительно к различным СУБД и проводят сравнительный анализ моделей.
2. **Физическое проектирование** - определение особенностей хранения данных, методов доступа и т.д.

Различие уровней представления данных на каждом этапе проектирования представлено в следующей таблице:

|  |  |
| --- | --- |
| **КОНЦЕПТУАЛЬНЫЙ УРОВЕНЬ*** сущности
* атрибуты
* связи
 | Представление аналитика |
| **ЛОГИЧЕСКИЙ УРОВЕНЬ*** записи
* элементы данных
* связи между записями
 | Представление программиста |
| **ФИЗИЧЕСКИЙ УРОВЕНЬ*** группирование данных
* индексы
* методы доступа
 | Представление администратора |

Литература:

* Тиори Т., Фрай Дж. Проектирование структур баз данных. М, 1985.
* Хаббард Дж. Автоматизированное проектирование баз данных. М, 1984.
* Бойко В.В., Савинков В.М. Проектирование баз данных информационных систем. М, 1989.

În capitolul anterior, am examinat pe scurt tehnicile de modelare funcțională care vă permit să izolați obiecte de informații primare, din care sunt apoi construite modele de date conceptuale și relaționale. Cu toate acestea, în cazul unui subiect destul de simplu, selectarea obiectelor informaționale poate fi efectuată fără analize funcționale. O modalitate de a proiecta o structură de baze de date relaționale în acest mod este descrisă în această secțiune și în următoarele secțiuni. Secțiunea [5.6](https://translate.googleusercontent.com/translate_c?depth=1&hl=ro&pto=aue&rurl=translate.google.com&sl=ru&sp=nmt4&tl=ro&u=http://www.mstu.edu.ru/study/materials/zelenkov/ch_5_6.html&usg=ALkJrhiwhh_6VKApALNoGV131ZtQ4rDdsg) discută un alt mod de a proiecta o structură relațională, bazată pe descompunerea relației universale.

**5.4 Modelare conceptuală Un exemplu de construire a unui model entitate-relație**

Un exemplu de construire a unui model entitate-relație a fost dat în secțiunea [2.1](https://translate.googleusercontent.com/translate_c?depth=1&hl=ro&pto=aue&rurl=translate.google.com&sl=ru&sp=nmt4&tl=ro&u=http://www.mstu.edu.ru/study/materials/zelenkov/ch_2_1.html&usg=ALkJrhihS8ylmMxL3e5nRPMKy5hvqaJumQ) , unde au fost introduse conceptele de bază ale acestui model. Aici vom analiza un alt exemplu legat de proiectarea bazei de date a **publicațiilor** , care a fost utilizată pentru instruirea practică în limbajul SQL.

Baza de date a **publicațiilor** ar trebui să stocheze informații despre publicațiile tipărite, precum și linkuri către resurse interesante de pe Internet. Atât acele, cât și alte surse de informații se vor ocupa de un singur subiect, și anume „baze de date”. Să încercăm să evidențiem entitățile care ne interesează și să stabilim relațiile dintre ele.

În primul rând, să ne ocupăm de conceptul de „ediție tipărită”. Ce este? Știm că obiectul „tipărit” este întruchipat sub forma unei cărți, care poate fi descris pe deplin folosind următoarele caracteristici: titlu, autor, anul publicării și editor (editor). Este posibil, pe baza acestui fapt, să introducem „cartea” entității și să definim caracteristicile numite ca atribute ale acesteia? Înainte de a face acest lucru, să aruncăm o privire mai atentă asupra relației dintre carte și caracteristicile sale:

* Un autor poate scrie mai multe cărți și, în același timp, o carte poate fi scrisă de mai mulți autori. În consecință, „carte“ și „autor“ , în acest caz , acționează ca entități diferite unite de *N: M* relație . Pentru a determina clasa de apartenență a entităților în legătură, observăm că nu există cărți fără autori, la fel cum nu există autori fără cărți. Aceasta înseamnă că fiecare entitate trebuie să aibă o clasă obligatorie de membru (relația cardinalitate *(1, N): (1, N)* ).
* În mod similar, un editor poate publica mai multe cărți simultan, dar fiecare carte este publicată într-un singur loc. Prin urmare, trebuie să introducem entitatea „editor“ asociat cu „cartea“ de un *1: N* relație . pentru că fiecare carte a fost publicată de cineva, clasa de apartenență a entității „editor” în acest sens va fi *(1,1)* , dar în același timp admitem stocarea informațiilor despre editori, ale căror cărți nu sunt încă în baza noastră de date. În consecință, clasa entității „carte” în acest sens *(0, N)* .
* În ceea ce privește caracteristicile cărții „titlu”, putem spune următoarele: de regulă, autorii care scriu pe un subiect încearcă să vină cu titluri originale pentru lucrările lor. Prin urmare, este sigur să presupunem că fiecare titlu este în mod necesar asociat cu o singură carte (și fiecare carte are un singur titlu). Prin urmare, „titlul” trebuie lăsat în lista atributelor „cărții”.
* Același raționament poate fi repetat și pentru caracterizarea „anul publicării”. O vom lăsa și în lista atributelor „cărții”.

Astfel, am stabilit că entitatea „carte” are două atribute „titlu” și „an de publicare”. După cum sa menționat deja, titlul va identifica cel mai probabil cartea dată, ceea ce nu se poate spune despre anul publicării. Prin urmare, vom declara atributul „title” (sau „book\_name”) ca cheie de entitate.

În ceea ce privește toți autorii posibili, ne interesează doar una dintre caracteristicile lor - numele. Prin urmare, entitatea „autor” are un singur atribut „nume\_autor”, care este cheia.

Situația cu entitatea „editor” este oarecum mai complicată. Aproape toți editorii mari au acum propriile pagini web, care pot conține informații utile pentru utilizatorii bazei de date proiectate. Prin urmare, trebuie să luați în considerare două caracteristici ale acestui obiect: „nume\_editor” și „URL” (localizator uniform al resurselor - un localizator universal al resurselor, cu care Internetul determină calea către o pagină web). Este clar că fiecare editor are un nume unic și o adresă URL unică, dar înainte de a le adăuga la lista de atribute, nu uitați că baza noastră de date trebuie să conțină și linkuri către alte resurse de Internet. Poate că, în timpul analizei ulterioare, va fi necesar să se introducă o entitate separată „URL”. Prin urmare, vom adăuga „publisher\_name” la lista atributelor entității ”editor ”și„ URL ”vor fi considerate un atribut al unei„ pagini web ”a unei entități separate asociată cu linkul„ editor ”*(1,1) :( 1,1)* .

Acum este momentul să abordăm obiectul resursei Internet. O putem descrie folosind conceptele „numele resursei”, „url”, „autor”. Având în vedere cu atenție conexiunile acestor concepte cu obiectul descris, se poate ajunge la concluzia că „nume\_resursă” și „adresă URL” sunt asociate în mod unic cu acesta, adică sunt atribute. În același timp, „autorul” este o entitate separată (o resursă poate avea mai mulți autori, iar un autor poate fi creatorul multor pagini web). pentru că am introdus deja entitatea „autor” mai devreme, vom defini pur și simplu caracteristicile conexiunii sale cu entitatea „resursă Internet”. Din cele de mai sus rezultă că aceste entități sunt unite de relația *n: m*, în același timp, este posibil ca autorul oricărei cărți să nu aibă propria sa pagină web, iar autorii unor resurse de internet nu își indică numele (adică putem spune oficial că aceste resurse nu au autori). Prin urmare, clasa de membru a ambelor entități va fi opțională.

Înainte de a declara modelul nostru gata, să verificăm de două ori definiția fiecărei entități. O analiză atentă va arăta că modelul construit are mai multe erori:

1. Entitatea autor are o clasă obligatorie de proprietate în raport cu entitatea de carte. Aceasta înseamnă că nu vom putea adăuga informații în baza de date despre persoana care și-a creat propriul site web, dar nu a scris nicio carte. Pentru a elimina această limitare, schimbați clasa de proprietate a entității „carte” din relația considerată „autor” - „carte” la opțional.
2. Când am analizat obiectul editorului, am presupus că entitatea paginii web ar putea fi combinată cu entitatea resursă Internet. Cu toate acestea, vedem că aceste entități au un set diferit de atribute, prin urmare, o astfel de unire nu poate fi realizată. Reamintim că altfel, s-a presupus că singurul atribut al entității „pagină web” va fi atașat atributelor entității „editor”. Cu toate acestea, nu vom face acest lucru, în secțiunea următoare vom vedea că folosind regulile pentru generarea relațiilor relaționale din modelul entitate-relație, vom obține același rezultat în ambele cazuri.

Modelul de relație entitate finalizat este prezentat în următoarea figură:



## 5.5.Правила порождения реляционных отношений из модели "сущность-связь"

### 5.5.1.Бинарные связи

|  |  |  |  |
| --- | --- | --- | --- |
| **Тип связи** | **Пример связи** | **Правило построения отношений** | **Отношения** |
| (1,1):(1,1) | http://www.mstu.edu.ru/study/materials/zelenkov/1111.gif | Требуется только одно отношение. Первичным ключом данного отношения может быть ключ любой из сущностей. | http://www.mstu.edu.ru/study/materials/zelenkov/r1111.gif |
| (1,1):(0,1)(1,1):(0,n) | http://www.mstu.edu.ru/study/materials/zelenkov/1101.gif | Для каждой сущности создается свое отношение, при этом ключи сущностей служат ключами соответствующих отношений. Кроме того, ключ сущности с обязательным классом принадлежности добавляется в качестве внешнего ключа в отношение, созданное для сущности с необязательным классом принадлежности. | http://www.mstu.edu.ru/study/materials/zelenkov/r1101.gif |
| (0,1):(0,1) | http://www.mstu.edu.ru/study/materials/zelenkov/0101.gif | Необходимо использовать три отношения: по одному для каждой сущности (ключи сущностей служат первичными ключами отношений) и одно отношение для связи. Отношение, выделенное для связи, имеет два атрибута - внешних ключа - по одному от каждой сущности. | http://www.mstu.edu.ru/study/materials/zelenkov/r0101.gif |
| (0,1):(0,n)(0,1):(1,n) | http://www.mstu.edu.ru/study/materials/zelenkov/010n.gif | Формируются три отношения: по одному для каждой сущности, причем ключ каждой сущности служит первичным ключом соттветствующего отношения, и одно отношение для связи. Отношение, выделенное для связи, имеет два атрибута - внешних ключа - по одному от каждой сущности. | http://www.mstu.edu.ru/study/materials/zelenkov/r010n.gif |
| n : m | http://www.mstu.edu.ru/study/materials/zelenkov/nm.gif | В этом случае всегда используются три отношения: по одному для каждой сущности, причем ключ каждой сущности служит первичным ключом соттветствующего отношения, и одно отношение для связи. Послденее отношение должно иметь среди своих атрибутов внешние ключи, по одному от каждой сущности. | http://www.mstu.edu.ru/study/materials/zelenkov/r010n.gif |

### 5.5.2.N - арные связи.

Общее правило: для представления n-сторонней связи всегда требуется n+1 отношение. Например, в случае трехсторонней связи необходимо использовать четыре отношения, по одному для каждой сущности (причем ключ сущности служит первичным ключом соответствующего отношения), и одно для связи. Отношение, порождаемой для связи, будет иметь среди своих атрибутов ключи от каждой сущности.



### 5.5.3.Иерархические связи.

К сожалению, надо признать, что реляционная модель мало подходит для отображения отношений наследования между сущностями (иерархических связей). Напомним, что в таких связях дочерние сущности наследуют все атрибуты родительской, и каждая из них обладает своим уникальным набором дополнительных атрибутов. В параграфе [2.2](http://www.mstu.edu.ru/study/materials/zelenkov/ch_2_2.html) приведен пример такой связи между родительской сущностью ЗАКАЗЧИК и дочерними - ЗАРУБЕЖНОЕ\_ПРЕДПРИЯТИЕ и ОТЕЧЕСТВЕННОЕ\_ПРЕДПРИЯТИЕ.

В этом случае возможны два варианта построения реляционных отношений. Согласно первому для иерархической структуры создается одно отношение, которое содержит атрибуты связи и всех сущностей. Для примера из параграфа [2.2](http://www.mstu.edu.ru/study/materials/zelenkov/ch_2_2.html) мы должны создать отношение ЗАКАЗЧИК(НАЦ\_ПРИНАДЛЕЖНОСТЬ, ВАЛЮТА, ЯЗЫК, ФОРМА\_СОБСТВЕННОСТИ). Недостаток такого способа - для каждого кортежа часть атрибутов всегда будет неопределена. Т.е. для отечественного предприятия всегда будут иметь значения NULL атрибуты ВАЛЮТА и ЯЗЫК, а для зарубежного атрибут ФОРМА\_СОБСТВЕННОСТИ. Более того, этот факт является требованием целостности сущности, следовательно для СУБД должны быть явно указаны несколько списков атрибутов (по числу дочерних сущностей), причем определенные значения могут быть присвоены только членам одного из них. Реляционная модель не поддерживает такого ограничения, на практике его реализуют с помощью триггеров.

По второму способу генерируется по одному отношению для каждой дочерней сущности. Каждое из этих отношений включает атрибуты родительской сущности и связи кроме атрибутов - дискриминантов т.е. ЗАРУБЕЖНОЕ\_ПРЕДПРИЯТИЕ(ВАЛЮТА, ЯЗЫК) и ОТЕЧЕСТВЕННОЕ\_ПРЕДПРИЯТИЕ( ФОРМА\_СОБСТВЕННОСТИ ). Недостатком данного способа является невозможность получить в одном запросе список всех заказчиков.

Оба описанных способа представлены на рисунке:



**Следует отметить, что построенные таким образом реляционные отношения, не являются окончательной схемой базы данных. Их необходимо проверить на избыточные функциональные зависимости и привести к NFBK или нормальной форме более высокого порядка.**

Применив все эти правила к модели "сущность-связь" базы данных **publications**, построенной в предыдущем параграфе, получим следующую реляционную структуру:



Синим цветом на диаграмме выделены первичные ключи, красным - внешние. Отношения, созданные для представления связей, обозначены серыми прямоугольниками, для сущностей - желтыми прямоугольниками.

Литература:

* Г.Джексон Проектирование реляционных баз данных для использования с микроЭВМ. М, Мир, 1991.

Следующая глава: [5.6.Проектирование реляционной базы данных на основе декомпозиции универсального отношения.](http://www.mstu.edu.ru/study/materials/zelenkov/ch_5_6.html)