32. Konceptuální modelování a návrh relační databáze.

(Zdroje: Opora IDS)

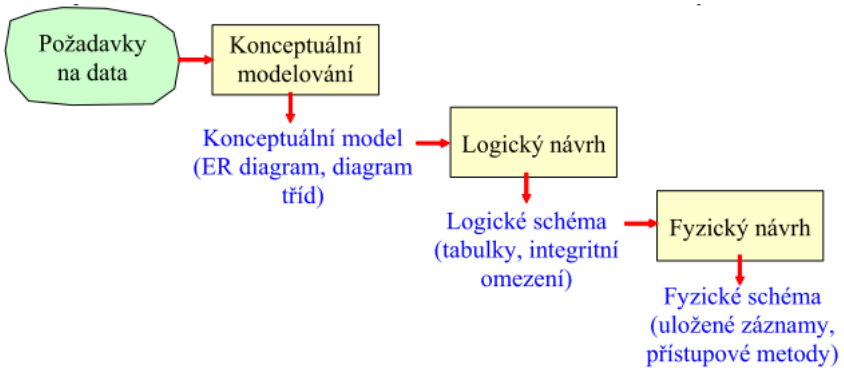
Konceptuální modelování (konceptuální návrh)

- při návrhu relační databáze patří do etapy analýzy požadavků
- je založeno na objektech (konceptech) daného souftwarového systému, který vyvíjíme. Pokud budeme modelovat např. systém pro školu, budeme tedy modelovat objekty reprezentující studenty, předměty, vztahy mezi nimi atd.
- jeho výsledkem je konceptuální model (ER diagram, diagram tříd)
- někdy se také nazývá *sémantické modelování*

Konceptuální vs. relační model dat:

Relační model dat popisuje obecnou strukturu, integritní omezení a operace nad daty v relační databázi bez ohledu na konkrétní aplikační doménu.

Naproti tomu konceptuální model modeluje data konkrétní aplikační domény, např. fakulty.



Obr. 3.1 Základní kroky návrhu relační databáze

Logický návrh

Jeho cílem je navrhnout z konceptuálního modelu (ER diagramu...) strukturu databáze. Je nutné aby databáze umožňovala reprezentaci všech potřebných informací, nebyla v ní redundance a měla jednoduchou kontrolu integritních omezení.

Jeho výsledkem je logické schéma databáze (*schéma relační DB*)

Fyzický návrh

Jeho cílem je navrhnout fyzické schéma uložení tabulek. Důležitou součástí organizace dat v databázi na fyzické úrovni je podpora SŘBD pro efektivní přístup k datům tabulek. Mezi tyto přístupové metody patří především indexování a hašování.

Pod fyzickým schématem si tedy můžete představit zejména informaci o umístění tabulek ve

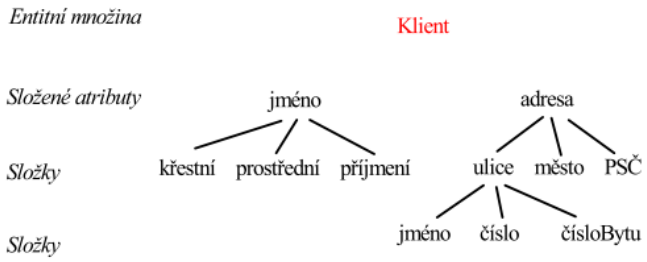
fyzické struktuře databáze a o přístupových metodách k záznamům tabulek.
Organizaci na fyzické úrovni relační model neurčuje, je to již záležitost jeho implementace.

Návrh relační databáze se tedy nejčastěji provádí v pořadí Vytvoření konceptuálního modelu (ER) a poté jeho transformací.
Přístup kdy se na začátku předpokládá,že všechna data jsou v jedné tabulce a poté se tabulka normalizuje, se nepoužívá neboť je značně neefektivní pro větší systémy.

ER diagram
-je výsledkem entitně-vztahového modelování
-Chápe modelovanou aplikační doménu jako množinu entit, mezi nimiž mohou existovat určité vztahy. Důležitým rysem ER modelu je, že popisuje data „v klidu“. Neukazuje, jaké operace budou s těmito daty prováděny.

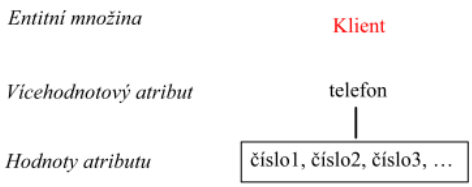
Entita
-je objekt reálného světa, který je rozlišitelný od jiných objektů a o kterém potřebujeme mít informace v databázi.
-Entity mohou být v určitých vztazích. (ale ER diagram modelu pouze to, že tento vztah existuje.)
- *Vztahová množina* - je množina vztahů téhož typu. V ER diagramu se značí úsečkou mezi entitami

Atributy
-Vlastnosti entit, jejichž hodnoty je potřeba ukládat do databáze.
-mohou být svojího typu: *Jednoduché* (nabývá pouze atomických=skalárních hodnot) a *složené* (skládá se z několika jiných atributů. Jednoduchých či složených)



Obr. 3.4 Složené atributy

-Atribut může být *jednohodnotový* (v každém okamžiku nabývá právě jedné hodnoty) a *vícehodnotový* (může současně nabývat několik hodnot)



Obr. 3.5 Vícehodnotový atribut

-vlastností atributu je i to, zda může být jeho hodnota prázdná (null). Jedná se o integritní omezení

Odvozený atribut
je takový atribut, jehož hodnota může být odvozena z jiných atributů a vztahů v DB. Např. věk klienta vypočitatelný z data narození.

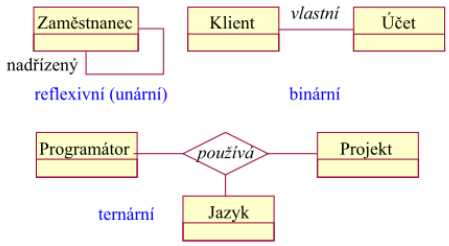
Primární klíč
atribut (může být i složený) jehož hodnota jednoznačně určuje danou entitu v rámci entitní množiny

Vlastnosti vztahů a vztahových množin

- pojmenování: vztahové množiny nemusí být pojmenovány. Pokud jej pojmenujeme volí se nejčastěji taková podoba aby připojením entitních množin vznikla jednoduchá věta.

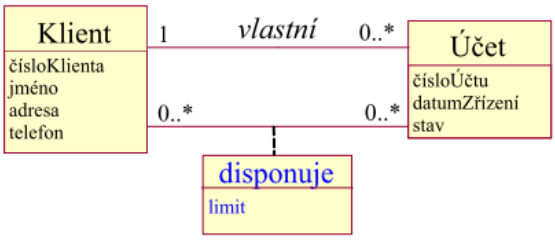
př. Klient vlastní účet.

- *stupeň vztahu*: počet entit vstupujících do vztahu.



Obr. 3.7 Stupeň vztahu

- *kardinalita*: jedná se o vlastnost “konce” vztahu. Udává, do kolika vztahů daného typu může jedna entita množiny na tomto konci vstupovat.
  typy kardinality
  -interval: 0..3 udává minimální a maximální počet vztahů daného typu
  -členství: minimální kardinalita (jedno číslo). Může jít být libovolné nezáporné číslo. Nejčastěji 0 (nepovinné členství) nebo 1(povinné členství)
  -maximální kardinalita: opět libovolné nezáporné číslo. Nejčastěji 1 nebo M (z *many*), které se v UML značí *
- atributy vztahů: Vztahy mohou mít podobně jako entity atributy. V ER diagramu se znázorňují jako obdélník asociovaný s úsečkou vztahu.

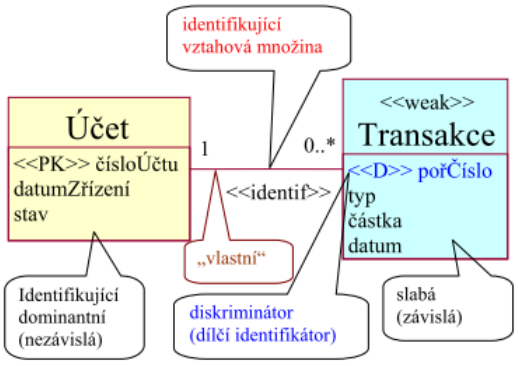


Obr. 3.10 Atributy vztahu

Rozšíření klasického ER diagramu

1. Slabé entitní množiny (weak entity set)

- entity závislé na jiných entitách. Nemohou existovat samostatně.
- obsahuje existenční závislost na jiné entitě, označované jako *identifikující* nebo *dominantní*
- tato závislost se projevuje tím, že identifikátor slabé entity je vždy složený a je tvořen identifikátorem silné entitní množiny a dalším atributem, který je unikátní v rámci entit závislých na stejné identifikující entitě. Ten se nazývá *diskriminátor* nebo *částečný identifikátor* (př. klient má na účtu transakce 1, 2 a 3. Číslo těchto transakcí je unikátní ve vztahu k jednomu klientovi-identifikátoru silné entity. Více uživatelů může mít své transakce číslovány také jako 1 a 2)
- každá slabá entitní množina je ve vztahu M:1 se svojí identifikující množinou. Vztah se většinou nepojmenovává (případně se používá "A vlastní B")
  - stereotypy: <<weak>> pro slabou entitní množinu, <<identif>> pro identifikující vztah a <<D>> pro diskriminátor

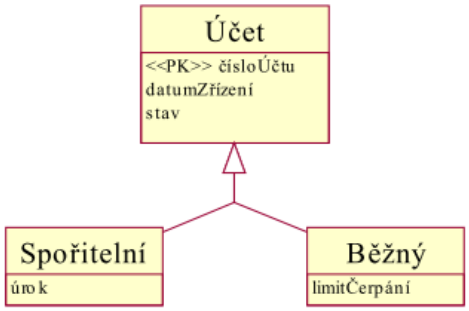


Obr. 3.12 Pojmy související se slabými entitními množinami

2. Generalizace a specifikace

-podobný princip jako dědičnost v OOP

-dědí se identifikátory(na obrázka by to bylo číslo účtu)



-Generalizace/specializace může být zpřesněna použitím dvou navzájem nezávislých charakteristik příslušnost a úplnost.

- *Příslušnost (membership)* - ·udává, zda mohou některé entity patřit do více než jedné specializační entitní množiny. Rozlišuje se příslušnost *disjunktní (disjoint)*, kdy jsou specializace disjunktní, a *překrývající se (overlapping)*, kdy nemusí být specializační entitní množiny disjunktní. (V obrázku jsou Disjunktní specifikace-účet je bud Spořitelní nebo Běžný)
- *Úplnost (completeness)* - udává, zda každá entita generalizující entitní množiny musí být zároveň entitou některé ze specializací. Příslušnost může být buď *úplná (total)* nebo *částečná (partial)*. V prvém případě musí každá entita generalizující entitní množiny spadat také do některé specializace, ve druhém mohou existovat entity, které nelze zařadit do některé ze specializací.

Návrh relační databáze

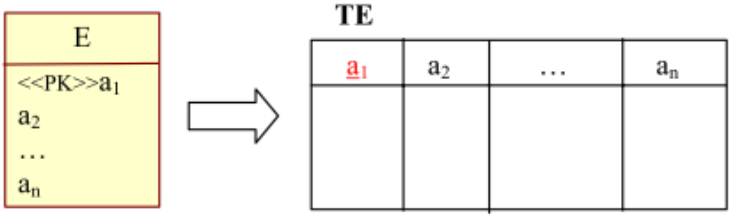
Redundance - opakování se souvisejících hodnot v jedné tabulce ve dvou a více sloupcích. Zvyšuje paměťovou náročnost, ale hlavně ztěžuje kontrolu integritních omezení.

Hlavní problémy chybného návrhu

- redundance
- složitá kontrola integritních omezí
- nemožnost reprezentovat určitou informaci

Pravidla transformace ER-diagramu na schéma relační DB

1. **odstranění složených a vícehodnotových atributů**
   -složený atribut rozdělíme na více atomických složek
   -vícehodnotový atribut- bud omezíme počet hodnot atributu (a vytvoříme tak např položky tabulky telefon1, telefon2 a telefon3). Nebo zavedeme novou entitní množinu (slabou nebo ji můžeme modelovat i jako silnou)
2. **Transformace silné entitních množin**
   -převedeme na tabulky. Sloupce tabulky odpovídají atributům entitní množiny, identifikátor se stane primárním klíčem. Jednotlivé řádky tabulky poté reprezentují jednotlivé entity daně entitní množiny



Obr. 3.35 Transformace silné entitní množiny

3. **Transformace binárního vztahu (2 entity)**
   -Vztahy s kardinalitou 1:1 převedeme tak, že rozšíříme jednu z tabulek o sloupec tabulku druhé (příapdně zkopírujeme i další sloupce)=sloučíme data do jedné tabulky
   -Vztahy s kardinalitou 1:M reprezentujeme v databázi tak, že rozšíříme tabulku pro entity, které mají na svém konci transformovaného vztahu kardinalitu M, o sloupec, který je cizím klíčem odkazujícím se na primární klíč druhé z obou tabulek. =dvě tabulky, PK jedné je cizím klíčem ve druhé (v té na které staně je M)
   -vztahy s kardinalitou M:M řešíme přidáním nové tabulky, která bude obsahovat jako cizí klíče PK obou připojených tabulek+atributy tohoto vztahu. Primární klíč je poté tvořen složeným klíčem obou cizích klíčů (případně +nějaký další atribut pokud kombinace dvou cizích klíčů není unikátní)
4. **Transformace ternární vztahové množiny**
   -je potřeba přidání vazební tabulky
   -Vazební tabulka bude obsahovat primární klíče všech tří tabulek, které zde budou cizími klíči a současně budou tyto tři cizí klíče tvořit primární klíč vazební tabulky.
5. **Transformace slabých entitních množin**
   -Slabou entitní množinu reprezentujeme v databázi tabulkou, která bude mít sloupce odpovídající jejím atributům a navíc bude mít sloupec cizího klíče, který se bude odkazovat na hodnoty primárního klíče tabulky reprezentující identifikující entitní množinu. Primární klíč tabulky bude složený a bude tvořen tímto cizím klíčem a diskriminátorem.
   Převod vztahu generalizace-specifikace
   Existuje více možností:

   -tabulka pro Nadtyp, a tabulka pro její specializaci (tj. pro každý Podtyp). Tabulky s podtypy, kromě svých atributů, budou mít navíc i sloupec s cizím klíčem do tabulky Nadtypu. Tento sloupec je PK v tabulce podtypu.
   -Tabulka pro každý Podtyp, která obsahuje zároven i atributy nadtypu
   -jedna tabulka, ve které je uloženo vše. Obsahuje sloupce pro všechny atributy nadtypu a všech podtypů. Primárním klíčem je identifikátor nadtypu. Rozlišení jednotlivých podtypů je testem prázdné hodnoty ve sloupci atributu, který je pro daný typ irelevantní nebo zavedením sloupce, který plní funkci diskriminátoru.
   -kombinace předchozích: Jedna tabulka pro Nadtyp a jedna společná tabulka pro všechny atributy podtypů. Rozlišení o který podtyp se jedná je stejné jako v předchozím případě.

Transformace diagramu tříd na schéma relační DB
-třídy definují i operace. Ty při návrhu DB ignorujeme (výjimkou jsou triggery apod.)
-objekty jsou definovány pomocí OID. Pokud není mezi atributy žádný použitelný jako PK, zavedeme další atribut.
-pro složené a vícehodnotové atributy použijeme buď metody popsané u ER, nebo použijeme typy BLOB (binární data) nebo CLOB (znaková)

Proces normalizace (asi už to v téhle otázce být nemusí!)

Proces normalizace umožňuje vyvarovat se: opakující se informace (redundance), nemožnosti reprezentovat určitou informaci, složité kontrola integritních omezení. Myšlenka normalizace se opírá o dva základní koncepty: Hierarchii tzv. normálních forem, které odrážejí kvalitu návrhu a postup dekompozice schématu relace, které vykazuje nedostatky.

Nejčastěji se uvádí 6 normálních forem:

- 1NF odráží základní požadavky relačního modelu dat z hlediska složitosti dat – atomické hodnoty jednoduchých atributů.
- 2NF, 3NF a BCNF - definují požadované vlastnosti z hlediska funkčních závislostí atributů
- 4NF, 5NF - definují požadované vlastnosti z hlediska vícehodnotových závislostí, resp. závislostí na spojení

1NF

Relace je v první normální formě, právě když všechny její jednoduché domény obsahují pouze atomické hodnoty

2NF

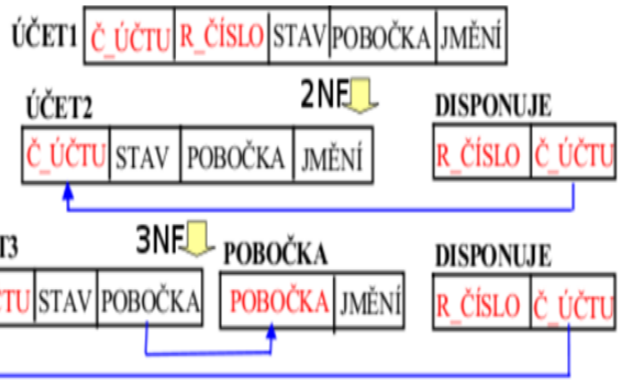
Schéma relace R je ve druhé normální formě, právě když je v 1NF a každý její neklíčový atribut, je plně funkčně závislý na každém kandidátním klíči relace R. (jednoduchý klíč určuje nezaměnitelně řádek tabulky)

3NF

Schéma relace R je ve třetí normální formě, právě když je ve 2NF a neexistuje žádný neklíčový atribut, který je tranzitivně závislý na některém kandidátním klíči relace R.

BCNF

Schéma relace R je v Boyce-Codově normální formě, právě když pro každou netriviální funkční závislost $X \rightarrow Y$ je X superklíčem. Superklíčem rozumíme každou nadmnožinu kandidátního klíče relace R. (všechny závislosti, které nejsou triviální, musí být dány závislostmi na celých kandidátních klíčích)



(Pozn. Nemusí dostačovat pro redundanci proto BCNF)