

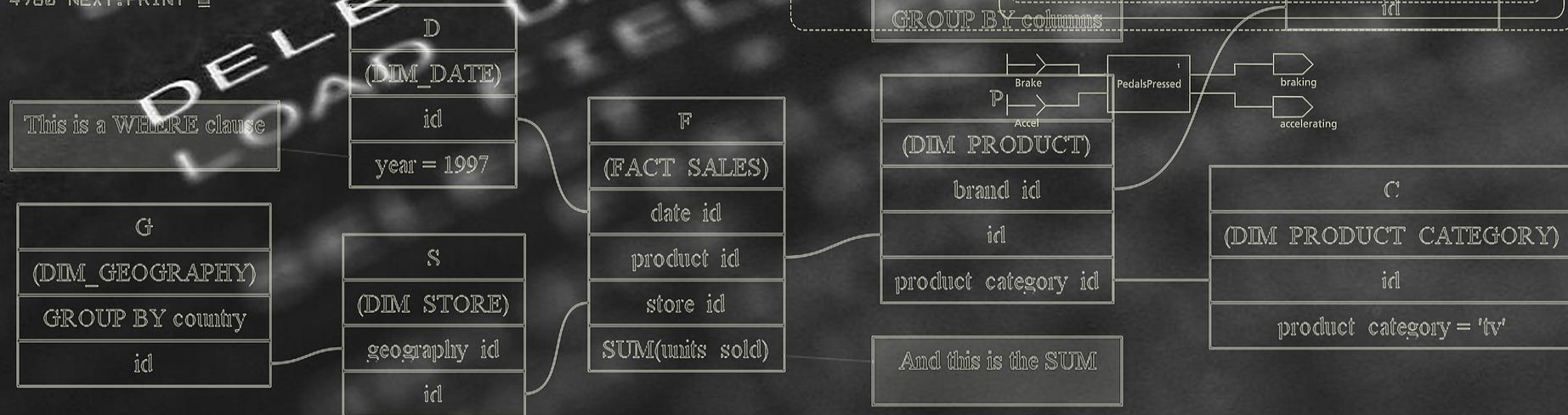
Design databáze

RNDr. Ondřej Zýka

```

4780 GOTO 5000
4790 :
4800 REM
4801 REM
4802 REM
4803 REM
4810 :
4820 PRINT
4825 W=V+1
4830 FOR X
4835 FOR I
4840 PRINT
4850 NEXT:
4860 PRINT
4870 FOR I
4880 IF MD
(I+1);:GOT
4890 PRINT
4900 NEXT
4910 PRINT
4920 FOR I
4925 PRINT
4930 IF MD
";:GOTO 4
4935 PRINT
4940 NEXT:PRINT"
4950 PRINT"#####";
4960 FOR I=2 TO 24 STEP 2
4965 PRINT"|";
4970 IF MD$(I+W-1)="  " THEN PRINT" "
MB$(I)" ";:GOTO 4980
4975 PRINT MB$(I);
4980 NEXT:PRINT" "

```



Návrh databáze (databázové části systému)

- Návrh má čtyři základní kroky
 - Shromáždění business požadavků
 - Vytvoření konceptuálního modelu
 - Vytvoření logického datového modelu
 - Vytvoření fyzického datového modelu (implementace)

- Při návrhu modelu databáze je nutné brát v úvahu
 - Jedná se o vývoj od začátku nebo o rozvoj stávajících systémů?
 - Nutnost začlenění okolních systémů (prostředí)
 - Vazba na logický model organizace
 - Vazba na existující datové modely

Shromáždění business požadavků

- Cíle
 - Pochopit business doménu
 - Porozumět potřebám a požadavkům zadavatelů a uživatelů
- Prostředky
 - Interview
 - Studium a dokumentace systémů
 - Spolupráce s experty v business oblasti
 - Informace o organizační struktuře a další dokumenty
 - Data assesment
 - Review stávajících systémů a procesů
- Výstup
 - Prioritizovaný seznam požadavků
 - Dokument popisující doménu (byznys slovník)
 - Data-flow diagram

Prioritizovaný seznam požadavků

- ID požadavku
 - Krátký popis
 - Podrobné vysvětlení
 - Oblast / systém
 - Vazba na další požadavky
 - Zadavatel
 - Priorita
-
- Funkční požadavky – popisují byznys funkce systému (Je vyžadováno zadání emailové adresy, Každý uživatel může mít přiřazen libovolný počet rolí, ...)
 - Nefunkční požadavky – popisují ostatní nároky na systém (Systém musí zvládnout práci 50 současně pracujících uživatelů, Administrace systému musí probíhat za běhu systému a nesmí snížit jeho výkonnost, ...)

Seznam požadavků - příklad

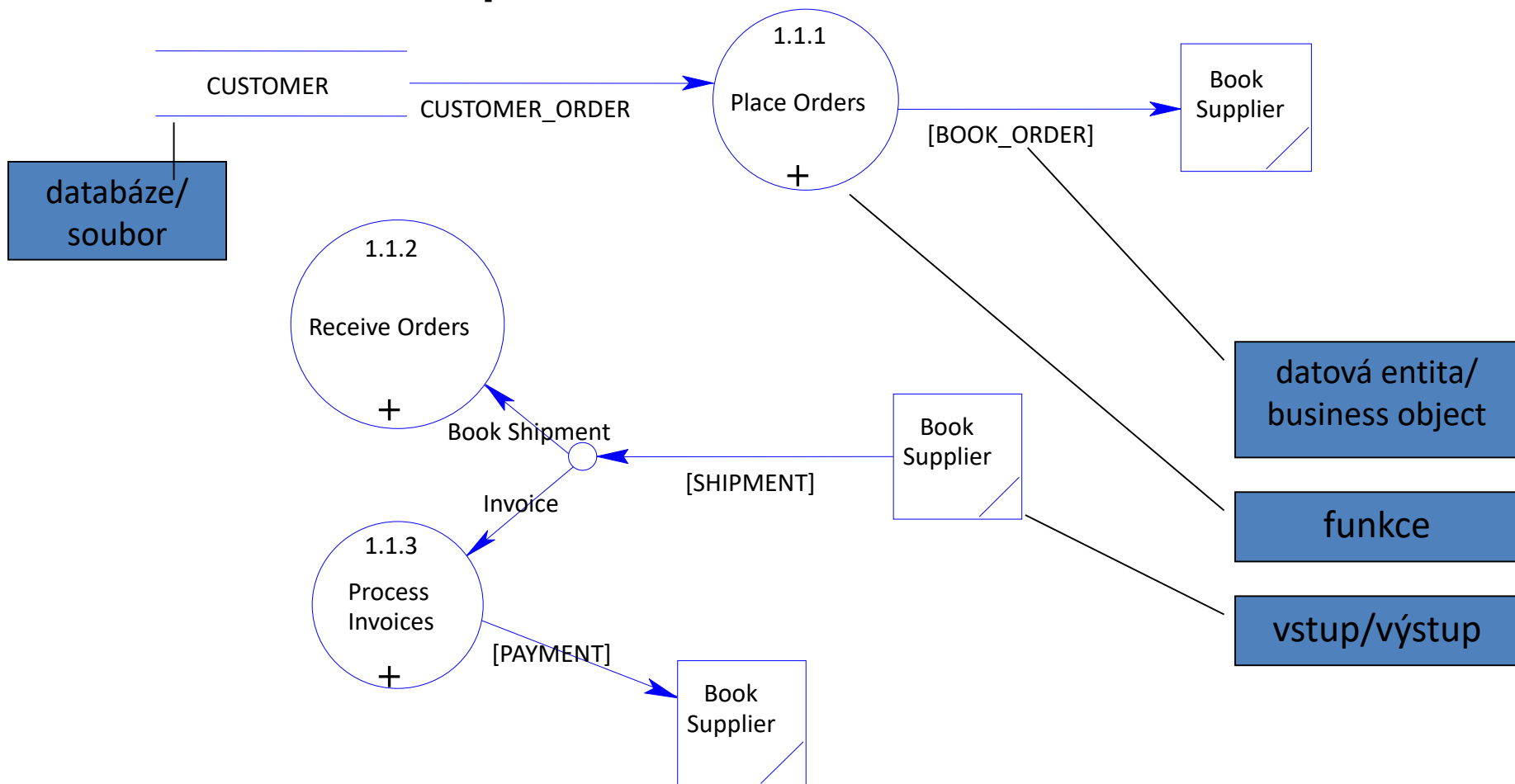
Business Functional Requirements								
#	Stream	Mainly impacted systems	Requirement	System description	Target system	Initiated by	Owner	Change Manager
68	F-REQ-03-068	N/A	Mass Activation - SIM Reservation GUI	This application has to be decommissioned after last customer is migrated out.	Extra process started probably from PDR. Generates given number of accounts in Arbor, PDR and EMA - network. Prepaid only.	Siebel CRM	KamiPetr	
69	F-REQ-03-067	N/A	Mass Activation GUI	This application has to be decommissioned after last customer is migrated out.	Extra process started probably from PDR. Generates given number of accounts in Arbor, PDR and EMA - network. Prepaid only.	N/A	KamiPetr	
70	F-REQ-03-068	N/A	MinDuration	This application has to be decommissioned after last customer is migrated out.	Corrects the call duration so that the first minute is charged completely even for short calls. Part of Arbor BP.	None	KamiPetr	
71	F-REQ-03-069	N/A	MMS Tracking Service	This application has to be decommissioned after last customer is migrated out.	Intranet GUI, CSR can see all application originated MMSes sent from VF to customer including binary ones. Data taken from extra databases built for MMP and Jbroker. 3rd parties to customer MMSes will be tracked even after hosting of MMSC.	None	KamiPetr	
72	F-REQ-03-070	N/A	MNP Interfaces	This application has to be decommissioned after last customer is migrated out.	Part of MNP, interface to Vagon	Siebel CRM	KamiPetr	
73	F-REQ-03-071	N/A	MNP Porting	This application has to be decommissioned after last customer is migrated out.	Part of MNP	Siebel CRM	KamiPetr	
74	F-REQ-03-072	N/A	MNP Proxy integration package	This application has to be decommissioned after last customer is migrated out.	MNP proxy integration. Package OSK\$MNP_API	N/A	KamiPetr	
75	F-REQ-03-073	N/A	MNP Proxy Vagon Interface	This application has to be decommissioned after last customer is migrated out.	Package OSK\$VOGON\$MNPPROXY	N/A	KamiPetr	
76	F-REQ-03-074	N/A	MOH	This application has to be decommissioned after last customer is migrated out.	Mass Order Handling. Stays until Arbor is replaced. Prepaid mass activation.	None	KamiPetr	
77	F-REQ-03-075	N/A	Motivation Report	This application has to be decommissioned after last customer is migrated out.	There is report of sales results... Part (from technical point of view) of Oskar Map Management. Data in BO_PROD1	Siebel CRM	KamiPetr	
78	F-REQ-03-076	N/A	MSF Discount	This application has to be decommissioned after last customer is migrated out.	belongs to Corporate preactivation. Infonet GUI?	None	KamiPetr	
79	F-REQ-03-077	N/A	MSISDN Recycling	This application has to be decommissioned after last customer is migrated out.	MSISDN recycling (CR4312)	Resource Management	KamiPetr	
80	F-REQ-03-078	N/A	MSISDN Selection	This application has to be decommissioned after last customer is migrated out.	The objective of MSISDN Selection application is to allow possibility to add new MSISDN on accounts for users in shops that are not allowed to have Arbor GUI. The application allows selecting MSISDN from postpaid, gold and data, fax and fax-box pools. The	None	KamiPetr	
81	F-REQ-03-079	N/A	MVPN Batch Creator	This application has to be decommissioned after last customer is migrated out.	Two parts of Batch Creator application, the BC web GUI and BC windows application. The first one is used for entering data from customer's forms into database and the second one generates from this data batch files for preprocessing in SMAS. Tool for mail	None	KamiPetr	
82	F-REQ-03-080	N/A	My Diary Admin	This application has to be decommissioned after last customer is migrated out.	Has been replaced by index	None	KamiPetr	
83	F-REQ-03-081	N/A	My Diary Admin '05	This application has to be decommissioned after last customer is migrated out.	Has been replaced by index	None	KamiPetr	
84	F-REQ-03-082	N/A	Nag SMS	This application has to be decommissioned after last customer is migrated out.	Sends information SMS that the service will change or has changed its status from paid to free and vice versa	N/A	KamiPetr	x
85	F-REQ-03-083	N/A	NAM Correction	This application has to be decommissioned after last customer is migrated out.	Part of supplementary activation. NAM is a param v HLR, setting the usage of GPRS and HSCDS.	None	KamiPetr	
86	F-REQ-03-084	N/A	Network Stories	This application has to be decommissioned after last customer is migrated out.	Web Presentations of articles about Vodafone Network part of Infonet. Application no more used, to be decommissioned.	None	KamiPetr	
	F-REQ-03-085	N/A	No Worries	This application has to be decommissioned after last customer is migrated out.	In Czech "Bez Obav". Once in 3 months calculates what would the usage and monthly fees cost under other Loaded tariffs. Of one of other Loaded tariffs costs less the difference	N/A	KamiPetr	

Data-flow diagram

- Data-flow diagram popisuje
 - S jakými daty se pracuje
 - Kdo data vytváří
 - Kdo a jak data zpracovává (modifikuje)
 - Kde jsou data uložena
 - Kdo data používá
 - Interface na úrovni dat
- Použití
 - Datové toky na různých úrovních granularity
 - Podnikové toky dat (obchodní procesy)
 - Toky dat na technické úrovni
 - Popisy ETL procesů
 - Popisy integračních procesů
- Kontroly diagramu
 - Všechna data jsou definována
 - Persistentní data jsou uložena
 - Každá data mají zdroj
 - Pro každá data existuje odběratel

Data-flow diagram - příklad

Decomposition Level 2



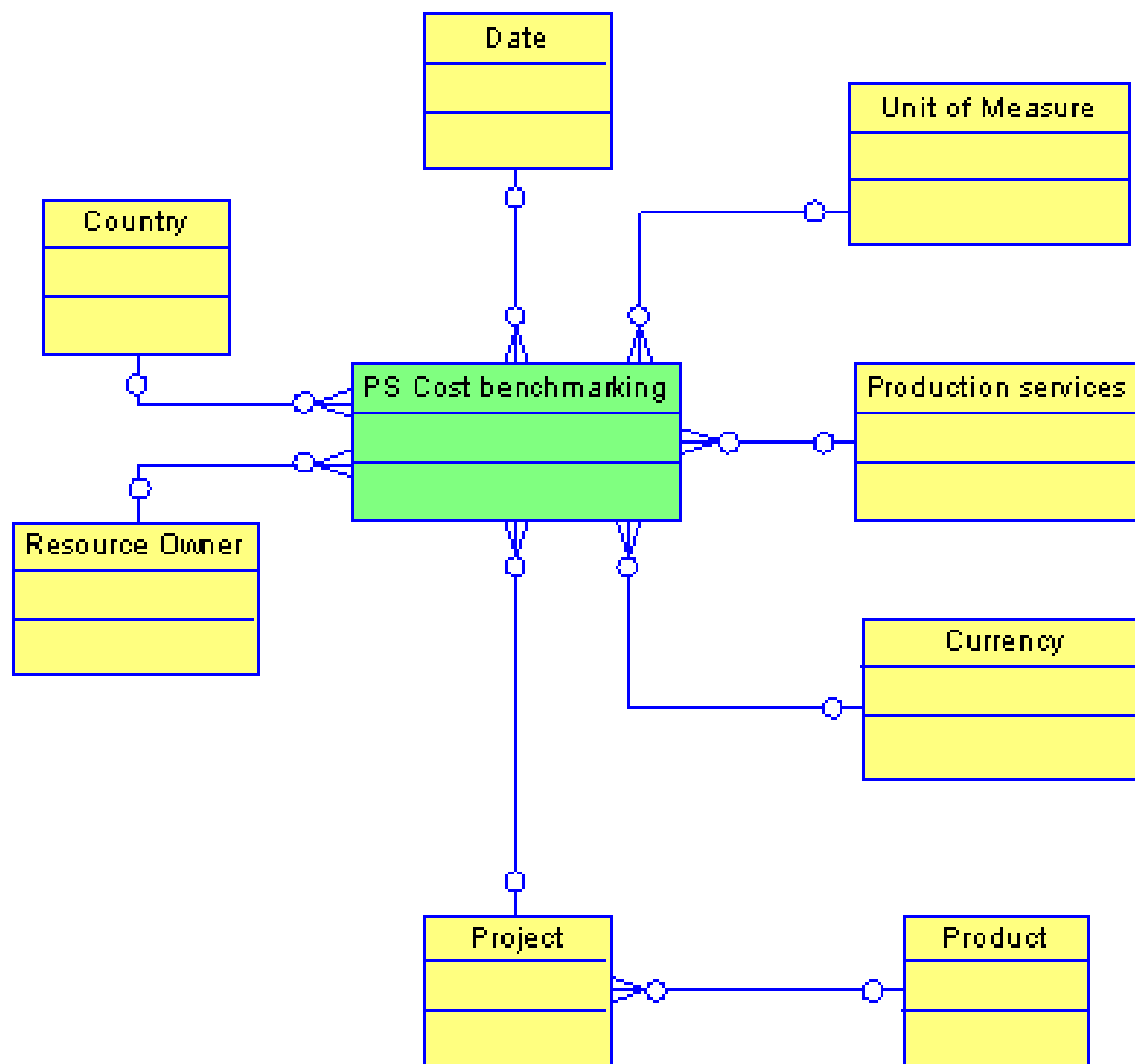
Vytvoření konceptuálního modelu

- Cíle
 - Ověřit pochopení zadání
 - Vytvořit podklad pro strukturovanou diskusi s uživateli o zadání a požadavcích
- Prostředky
 - Zpracování byznys požadavků
 - Diskuse s uživateli
- Výstup
 - Entity Relationship diagram (ER diagram)

ER diagram

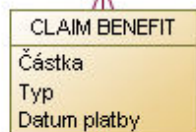
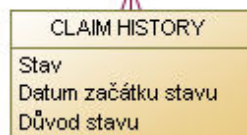
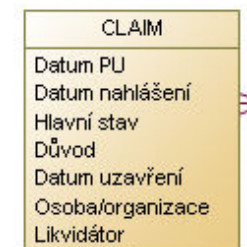
- ER diagram obsahuje
 - Entity
 - Určení slabých a silných (nezávislých a závislých) entit
 - Atributy entit
 - U atributů jejich hodnoty a vlastnosti
 - U entit jejich potenciální klíče (jak uživatelé identifikují entity, identifikátory)
 - Identifikace vazeb (relací) mezi entitami
 - Kardinalita relace
 - Jméno relace
 - Popis (role)
- Formální ověření E/R diagramu
 - Mezi každými dvěma entitami je maximálně jedna relace.
 - Neexistuje cyklická závislost.
 - Entity s relací typu 1:1 zřejmě budou tvořit pouze jednu entitu.
 - Žádná entita nemá atribut, který je kandidátním klíčem jiné entity.
 - Nepřímé relace jsou asi zbytečné.

ER diagram - příklady

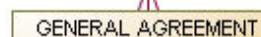
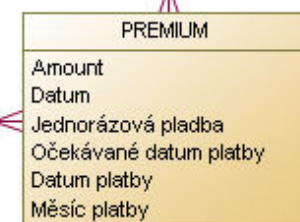
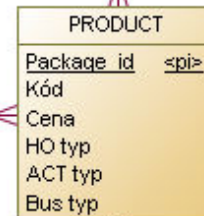
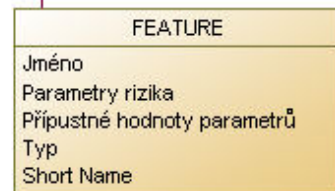
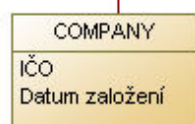
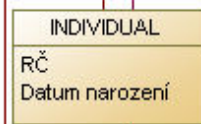
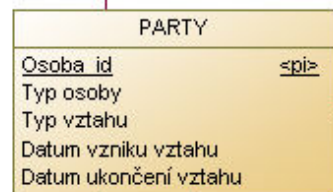
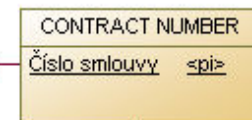
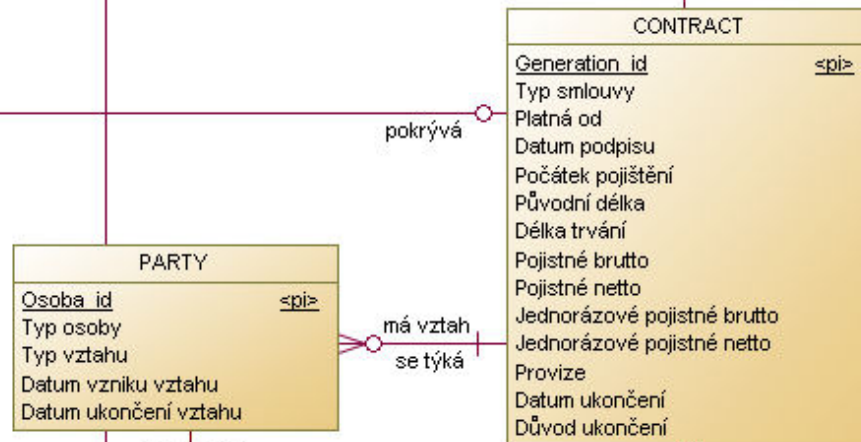
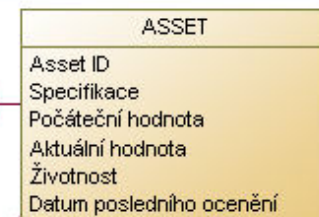


Škodné události a likvidace

Pojistka z pohledu koncového zákazníka



Finanční účetnictví



je pokryta

odpovídá

upřesňuje
je upřesněn

je upřesněn
upřesňuje

pokrývá

je pokryt
pokrývá

má vztah
se týká

náleží
má

zapláceno

obsahuje

je součástí

za

je součástí
skládá se

placen

za

obsahuje

je obsažen

Notace

Existuje mnoho standardů a „standardů“.

Většinou spojeno s konkrétním nástrojem obsahující rozšíření a specifická použití.

Na konkrétních projektech často přetěžováno podle konkrétních specifik a zvyklostí.

- **UML** – notace dle standardu UML, použita ve všechny nástroje podporující UML.
- **Information Engineering** – standard používaný v mnoha nástrojích. Existuje několik verzí. Obecně, entity jsou obdélníky a relace jsou linky s různými zakončeními.
- **IDEF1X** – standardní notace pro modelování relací a entit. Symboly označují kombinaci volitelnosti a kardinality entity.
- **Barker** – Vytvořena Richardem Barkerem. Používaná zejména case nástroji Oracle. Speciální notace pro dědičnost, vlastní notace pro násobnost a speciální značky pro atributy.
- **Filtered IE** – pouze v Embarcadero. Nezobrazuje cizí klíče.
- **Entity/Relationship** – Sybase specific, Entity/Relationship je speciální verze IE notace.
- **Merise** – používá asociace místo relací.
- **Crow's Feed** – Jedna z verzí IE notace, tuto notaci používá FSLDM.

Notace entit a atributů

- Většina prvků notace je snadno identifikovatelná
- Přesto je vždy nutné se dohodnout, jak je který symbol v daném projektu chápán.

PERSON		
<u>Person ID</u>	<pi>	<M>
CURRENT FIRST NAME		<M>
CURRENT MIDDLE NAME		
CURRENT LAST NAME		<M>
BIRTH DATE		
GENDER		
IDENTIFICATION NUMBER		

E/R

PERSON		
<u>Person ID</u>	<pi>	<M>
CURRENT FIRST NAME		<M>
CURRENT MIDDLE NAME		
CURRENT LAST NAME		<M>
BIRTH DATE		
GENDER		
IDENTIFICATION NUMBER		

IDEF1X

PERSON	
#	<u>Person ID</u>
*	CURRENT FIRST NAME
o	CURRENT MIDDLE NAME
*	CURRENT LAST NAME
o	BIRTH DATE
o	GENDER
o	IDENTIFICATION NUMBER

Barker

Notace - příklady

Notation	Information Engineering	Barker Notation	IDEF1X	UML
<u>Multiplicities:</u>				
- Zero or one				
- One only				
- Zero or more				
- One or more				
- Specific range	N/A substituted by	N/A	N/A	
<u>Associations:</u>				
Labels				
Entity roles	N/A	N/A	N/A	
Subtyping				
Aggregation				
Composition				

Allowed multiplicities

Not Used

Used

Not Used because not needed

Pravidla pro čtení



Each LINE_ITEM may part of one or more ORDER.
Each ORDER must composed of one and only one LINE_ITEM.



Each LINE_ITEM must part of one or more ORDER.
Each ORDER must composed of one and only one LINE_ITEM.

Pravidla pro čtení



Each LINE_ITEM may part of one or more ORDER.
Each ORDER may composed of at most one LINE_ITEM.



Each LINE_ITEM must part of one or more ORDER.
Each ORDER may composed of at most one LINE_ITEM.

Pravidla pro čtení

Entity 1	Entity 2
LINE_ITEM	ORDER

part of composed of

General Cardinalities Definition Rules

Each LINE_ITEM must part of one or more ORDER.
Each ORDER must composed of one and only one LINE_ITEM.

Cardinalities

☐ One - one ☒ One - many ☐ Many - one ☐ Many - many

Dominant role: <None>

LINE_ITEM to ORDER

Role name: part of

☐ Dependent ☒ Mandatory Cardinality: 1,n

ORDER to LINE_ITEM

Role name: composed of

☒ Dependent ☒ Mandatory Cardinality: 1,1

Vytvoření logického modelu

- Cíle
 - Vytvořit platformově nezávislý logický datový model
 - Aplikovat obecně uznávané postupy a pravidla pro relační datový model
- Postup
 - Rozhodnout o způsobu reprezentace dědičnosti (subtypů) – viz přednáška Patterny
 - Rozhodnout, jak se bude pracovat se složitými atributy
 - Rozhodnout, jak se bude pracovat s atributy nabývajícími více hodnot
 - Převést entity na tabulky
 - Rozhodnout o použití vhodných patternů – viz přednáška Patterny
 - Vybrat primární klíče
 - Převést binární relace (závislosti) typu 1:n na cizí klíče
 - Vyřešit n-ární relace a relace typu n:n
 - Provést normalizaci modelu
- Výstup
 - Logický datový model

Primární klíč

- Identifikátor entity – atribut (atributy), podle kterých uživatelé identifikují jednotlivé instance entity.
- Potencionální primární klíč – sloupce tabulky, podle kterých je možné identifikovat jednotlivé řádky.
- Primární klíč – jeden z potencionálních primárních klíčů, přenáší se jako cizí klíč pro implementaci vazby 1:n.
- Constraint Primary Key – identifikace primárního klíče v databázi. Většinou vynucuje index nad primárním klíčem a not null hodnoty.
- Constraint Foreign Key – implementace vazby 1:n, kdy se přenáší primární klíč hlavní tabulky do závislé tabulky.
- Primary index – některé databáze vyžadují definici primárního indexu určujícího fyzické rozložení dat. Nemusí být shodný s primárním klíčem, nemusí být unikátní, může obsahovat null hodnoty.

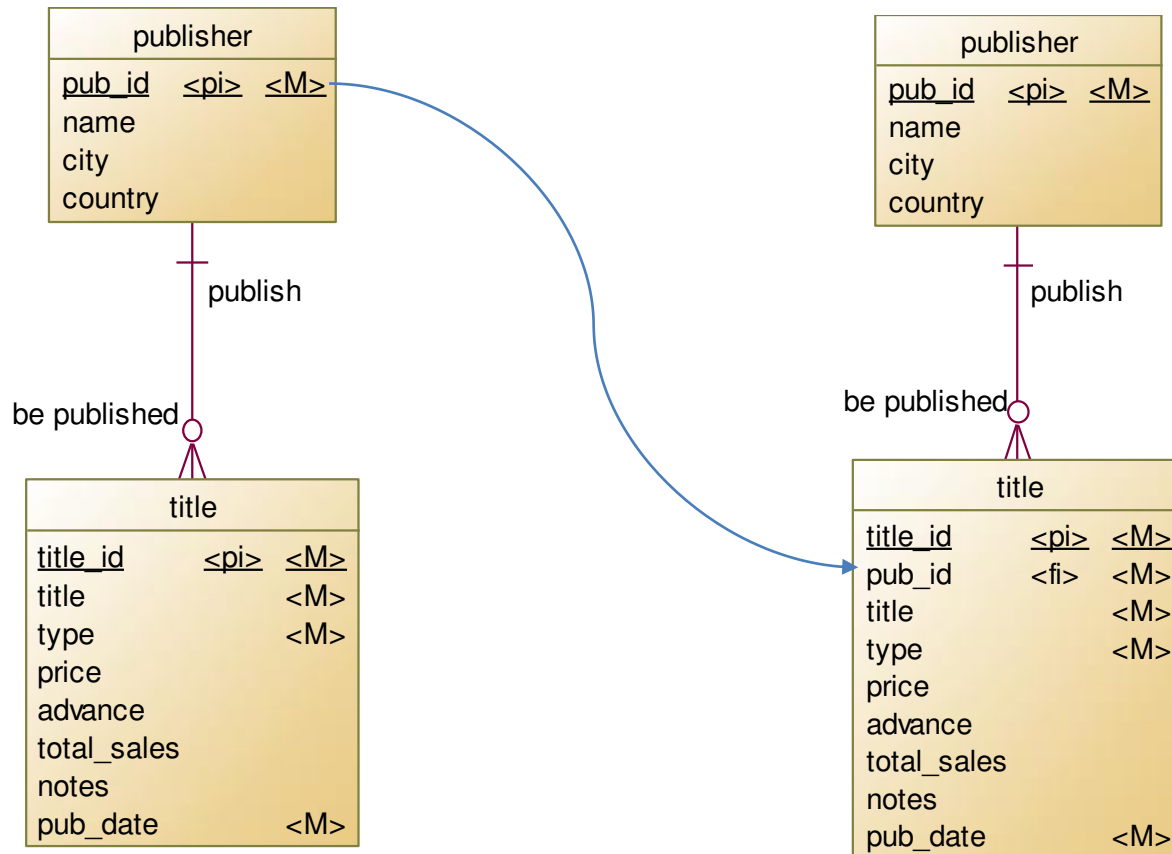
Výběr primárního klíče

- Z možných potencionálních primárních klíčů vybrat nejvhodnější, popřípadě vytvořit nový umělý primární klíč.
- Kritéria výběru primárního klíče z potenciálních primárních klíčů:
 - musí mít vždy definovanou hodnotu (not null),
 - musí mít stálou hodnotu (během celého životního cyklu řádku),
 - musí být co možná nejmenší,
 - nesmí obsahovat žádné zakódované informace,
 - musí být přístupný pro všechny uživatele.
- Vytvoření nového umělého klíče
 - Výhody
 - Snadná implementace rozhraní
 - Vytváří uniformní řešení
 - Součást některých patternů
 - Standard v objektovém programování
 - Nevýhody
 - Přidává sloupec do tabulky (s indexem)
 - Hodnoty nemají význam pro uživatele

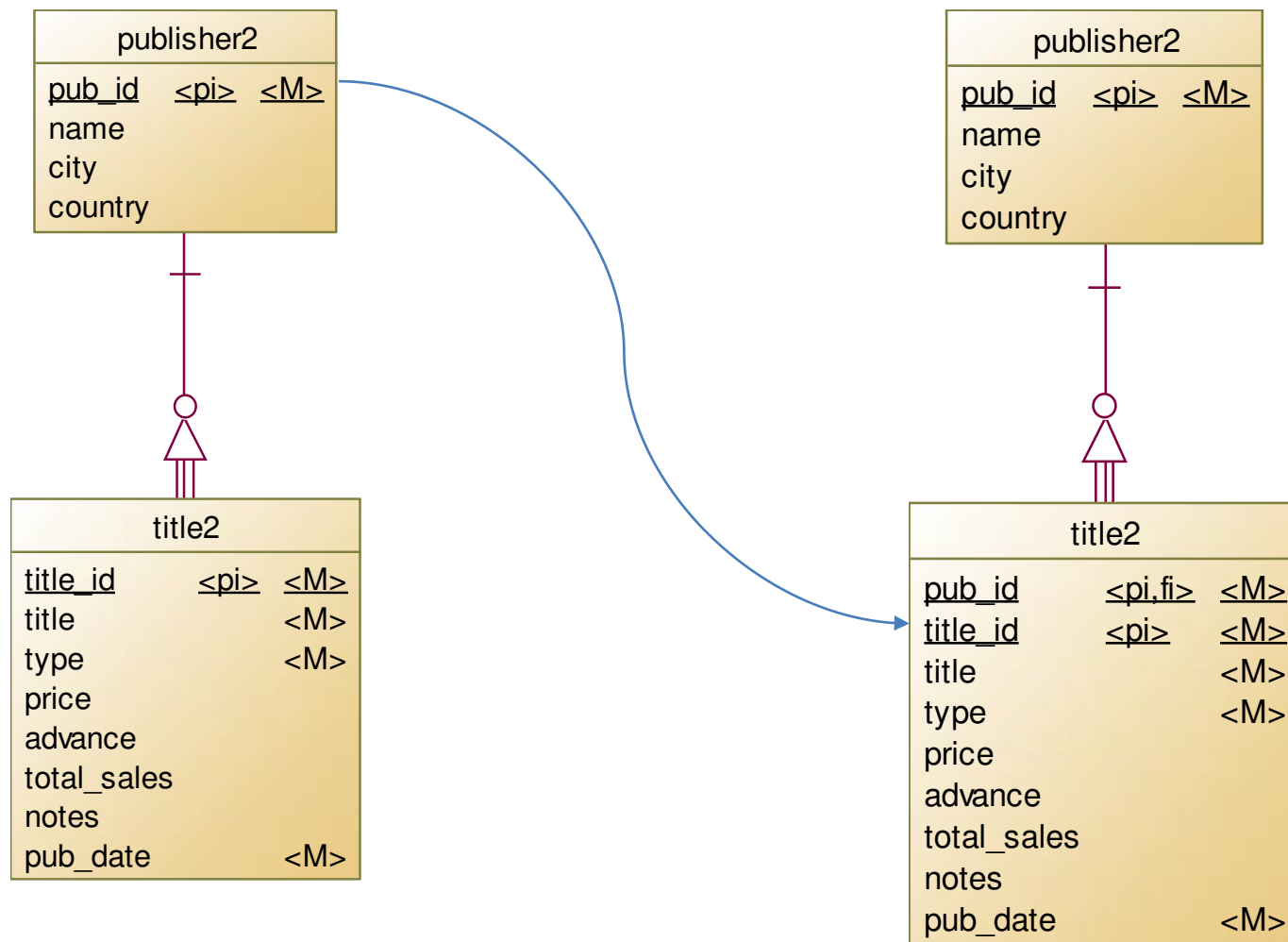
Vazba na objektové modelování

- Object-relational impedance mismatch
- Declarative vs. imperative interfaces
 - RM – data jako interface
- Schema bound
 - RM – Sloupec k jedné tabulce, tabulka do schématu, OOM – dědičnost objektů
- Access rules
 - RM – relační algebra, OOM – volnější a složitější konstrukty
- Relationship between nouns and actions
 - OOM - úzká vazba mezi objekty a operacemi
- Uniqueness observation
 - RM – identifikace na základě klíče s jasným obsahem
- Normalization
 - OOM – nepoužívá se normalizace
- Schema inheritance
 - RM – nepoužívá se
- Structure vs. Behaviour
 - OOM – údržba, srozumitelnost, upravovatelnost, rozšiřitelnost, reuse, RM – logická integrita, efektivita, fault-tolerance
- Set vs. graph relationships

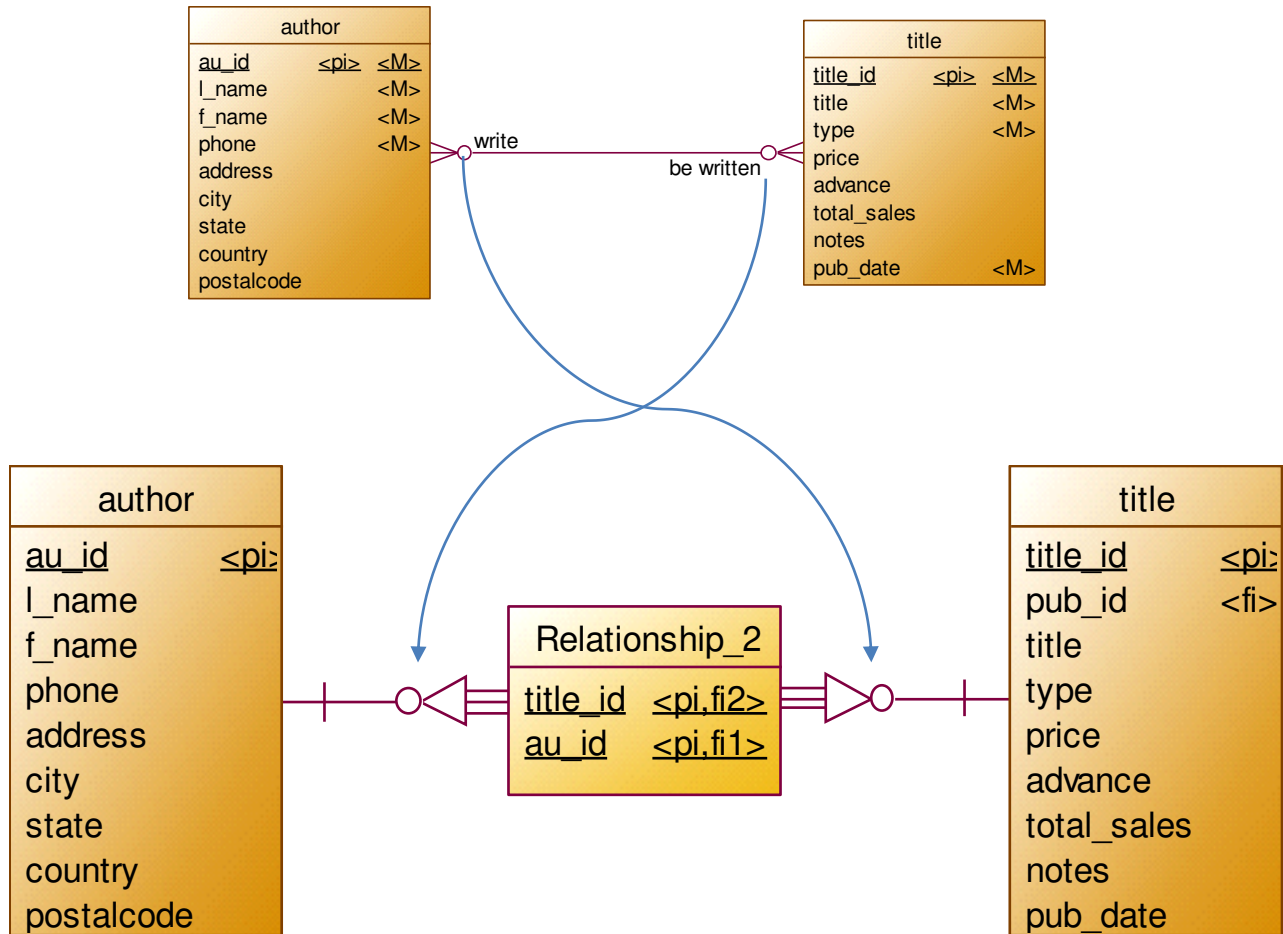
Převod binární relace 1:N na cizí klíč



Převod binární relace závislosti na cizí klíč



Převod relace typu n:n na tabulky



Normalizace modelu

- První normální forma
 - Tabulka je v první normální formě, když každý sloupec obsahuje právě jednu atomickou hodnotu, která již nemá vnitřní strukturu z pohledu uživatelů.
 - Uživatelská (byznys) pohled je podstatný pro rozhodnutí o první normální formě.
- Funkční závislost
 - Sloupec B je funkčně závislý na sloupci (nebo množině sloupců) A ($A \rightarrow B$), když hodnota ve sloupci B je určena hodnotou ve sloupci (nebo množině sloupců) A.
- Druhá normální forma
 - Platí první normální forma
 - Žádný sloupec mimo potenciaální primární klíč není funkčně závislý na podmnožině potenciačního primárního klíče.
- Třetí normální forma
 - Platí druhá normální forma
 - Každý sloupec mimo potenciaální primární klíč je funkčně závislý jenom na potenciačním primárním klíči.
- Boyce Codd (BCNF) normální forma
 - Platí třetí normální forma
 - Pokud je sloupec funkčně závislý, tak je funkčně závislý na primárním klíči.

Vytvoření fyzického modelu

■ Cíle

- Vytvořit fyzický model s ohledem na specifika aplikace a použitý typ databáze, použitý hardware

■ Postup

- Pojmenovat tabulky a sloupce (jmenné konvence)
- Vybrat datové typy
- Vytvořit procesní matici
- Revidovat strukturu tabulek
 - Denormalizace
 - Uložení redundantních dat
 - Spojení tabulek
- Revidovat rozhodnutí o primárním klíči
- Definovat primární indexy
- Rozhodnout o implementaci business pravidel - constraints
- Definovat indexy, rozhodnout o použití partitions
- Definovat fyzické uložení tabulek

■ Výstup

- Fyzický datový model
- Implementační skripty

Pojmenovat tabulky a sloupce (jmenné konvence)

- Co je třeba brát v úvahu:
 - Server může nebo nemusí rozlišovat velká a malá písmena
 - Server může podporovat jména s nestandardními znaky
 - Omezení délky jména tabulek objektů (!!!! Oracle stále 30 znaků)
 - Omezení na jmenné prostory (tabulky, view, indexy, procedury, ...), které objekty se mají jmenovat stejně
- Porozumění modelu
 - Jména podle typů objektů - datové tabulky, číselníky, logy, uživatelské tabulky (např. prefixy)
 - Související objekty – jména indexů na tabulce, primárních klíčů, constraintů, triggerů, ... (např. postfixy)
 - Standardní jména sloupců – id, name, type, comment, ... (např. přidávat jméno tabulky)
 - Natural join

Datové typy – co je nutné brát v úvahu

■ Domains

- Jednoduché uživatelsky definované datové typy
 - + lepší porozumění
 - + zajištění konzistence
 - - nároky na údržbu
- Strukturované uživatelsky definované datové typy speciální datové typy (XML, region, ...)
 - - nároky na vývoj
 - - špatná podpora klientských nástrojů
 - - špatná přenositelnost

■ Char, varchar, nvarchar, ...

- Porovnání char a varchar řetězců
- Null hodnota a prázdný řetězec
- Práce s národními znaky ve varchar a nvarchar řetězcích
 - vazba na použité klienty a aplikace
- Velikost varchar řetězců, nutnost použití typů text

■ Numerické datové typy

- int, tinyint, bigint, numeric(p,s), ...
- Velikost a přesnost, chování v extrémních případech
- Co znamená typ number (38,-65)?
- Jak malá čísla se dají zapsat do datového typu float nebo double precision

Datové typy – co je nutné brát v úvahu

- Datum – rozsah, přesnost, způsob práce
 - Timestamp – je to čas nebo není
 - Nutnost rozlišit hodnotu zapsanou v databázi a identifikaci časového okamžiku v reálném světě
 - Překlad času podle klienta
- Binary, image, text, memo, ...
 - Omezení operací nad velkými datovými typy (Indexy, vyhledávání, spojování)
 - Specifické požadavky na uložení dat
 - Často plýtvání místem v databázi
- Boolean - raději

```
"column_name" CHAR(1) default 'A' not null constraint  
CKC_check_name check ("column_name" in ('A','Y'))
```
- Implementace Identity, Autoincrement
- NULL, Not null, Default value
 - Vazba na výkonnost databáze

Procesní matice

- Slouží k identifikaci nejdůležitějších operací nad daty
 - Identifikace hlavních procesů
 - Select/Insert/Update/Delete
 - Zpracovávané množství dat
 - Typy výběrů dat
 - Nejčastěji společně používaná data
 - Požadavky na odezvu
 - Požadavky na průchodnost
- Často stačí jenom kód nebo pseudokód pro hlavní operace

Procesní matice - příklad

Process	tableA.col1	tableA.col2	tableA.col3	tableB.col1	tableB.col2	tableX ...	Order By	Frequency	How Critical	Service Level	Location	Algorithm or Qualification
P1	S(*)	S		S	S		A.col1	1000/day	3	< 1 sec	London	sum(B.col2)
P2				S(*)	S			50,000/day	1	<1 sec	Paris	B.col1 = "constant"
P3	S 40	S(*) 40		S(=)	S		A.col2	99,000/day	1	< 2 sec	Boston	A.col2 like "B%"
	S 30	S(*) 30		S(=)	S		A.col2	44,000/day	1	< 4 sec	Paris	A.col2 like "P%"
...												

Revize struktury tabulek

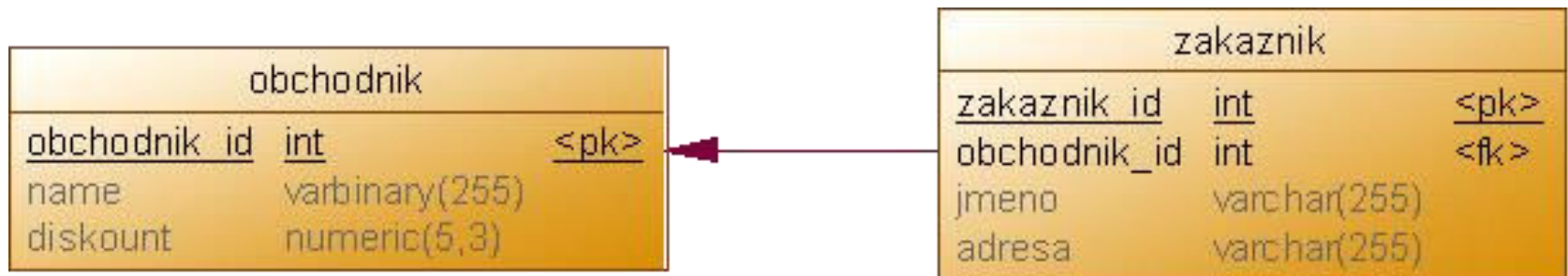
- Cíle:
 - Minimalizovat velikost
 - Použít optimální přístupové metody
- Standardní tabulka
 - Insert, Delete, Update, Scan, Index
- Index-organized tables (Clustered index)
 - + menší
 - + rychlejší přístup po indexu
 - - náročnější insert, delete (update)
 - Oracle – novější implementace, podpora 7x24
- Column organized tables

Implementace business pravidel

- Domain integrity - Check
 - Na úrovni sloupce
 - Null/Not null
 - Default
 - Check (format)
 - Na úrovni řádku
- Entity integrity - primary key
 - implementováno unikátním indexem na not null sloupcích
 - záznam v katalogu
- Unikátnost hodnot
 - implementováno unikátním indexem
- Pravidla implementovaná pomocí triggerů

Referenční integrita

- Možné implementace referenční integrity
 - Deklarativní definice
 - Použití triggerů
 - Použití uložených procedur
 - Kód aplikace
- Co se stane když
 - Primární klíč se přidá/změní/zruší
 - Cizí klíč se přidá/změní/zruší



Typy referenčních integrit

	Constraint	Cascade update / delete	Cascade Null	Cascade Default	Automatic insert	Bez omezení	Restriktivní
Insert PK	Zachování PK	Zachování PK	Zachování PK	Zachování PK	Zachování PK	Zachování PK	Zachování PK
Insert FK	Pouze existující hodnoty PK	Pouze existující hodnoty PK	Existující hodnoty PK nebo Null	Existující hodnoty PK nebo Default	Existující hodnoty nebo přidá řádek s PK	Bez omezení (sirotci)	Pouze existující hodnoty PK
Update PK	Pouze pokud nemá vazbu	Změní i odpovídající FK	Odpovídající FK změní na Null	Odpovídající FK změní na Default	Pouze pokud nemá vazbu	Bez omezení (sirotci)	Není povoleno
Update FK	Jen na existující hodnoty	Jen na existující hodnoty	Jen na existující hodnoty PK nebo Null	Jen na existující hodnoty PK nebo Default	Existující hodnoty nebo přidá řádek s PK	Bez omezení (sirotci)	Jen na existující hodnoty
Delete PK	Pouze pokud nemá vazbu	Delete všech řádků s odpovídajícím FK	Odpovídající FK změní na Null	Odpovídající FK změní na Default	Pouze pokud nemá vazbu	Bez omezení (sirotci)	Není povoleno
Delete FK	Bez omezení	Bez omezení	Bez omezení	Bez omezení	Bez omezení	Bez omezení	Bez omezení

Důsledky použití integritních omezení

- + Zaručují konzistentní model

- Zvyšují výpočetní složitost

 - Tabulka s mnoha cizími klíči – při každé změně kontrola všech hodnot
I v případě, kdy nedochází ke změnám (Oracle not null)

- Komplikují údržbu

 - Nutnost povolení/zakázání ověřování integritních omezení při administrativních operacích
 - Nahrávání historických dat
 - Řešení chybových stavů

Denormalizace

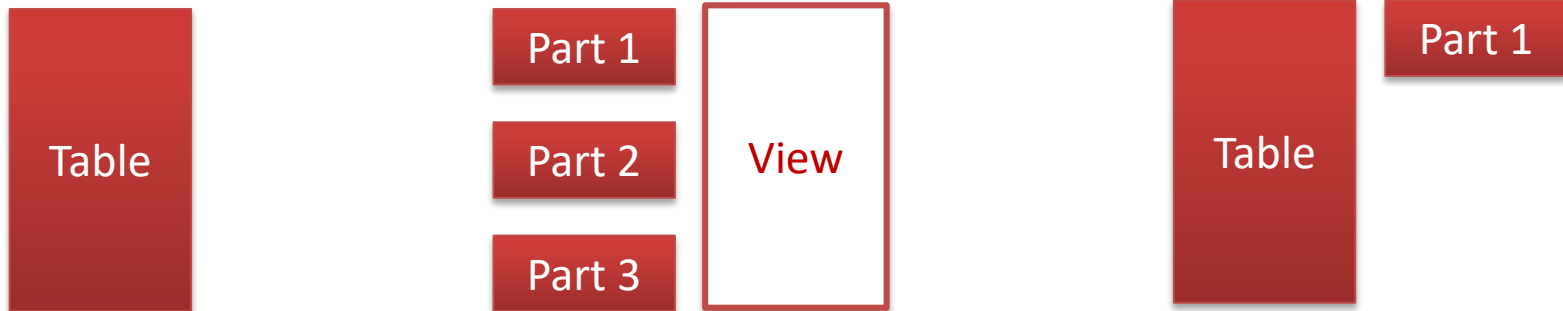
- Partitioning
 - Horizontální
 - Vertikální
- Uložení vypočtených hodnot
- Eliminace nákladných joinů

Horizontální rozdělení tabulek

- Přístupy pouze na část tabulky
- Příklady:
 - Aktivní a neaktivní položky
 - Historické záznamy
- Možnosti
 - Rozdělení tabulek
 - Přidání tabulky (duplicitní záznamy)
 - Partitioning
- Synchronizace
 - Table partitioning
 - Triggery
 - Aplikační logika

Rozdělení tabulky

Rozdělení tabulky



Výhody

- práce s menším množstvím dat
- méně problémů se zamykáním
- lepší řízení indexů
- možnost detailní optimalizace

Nevýhody

- nutnost synchronizace – triggerů, aplikační logika
- Náročnější údržba

Partitioning

- Transparentní z pohledu aplikace
- Rozdělení dle daného rozsahu nebo hodnot
- Dynamicky podle hodnot
 - Dynamicky vytvářeno pro každý měsíc
 - Nejčastější použití
- Podle hash klíče (určuje se pouze počet partitions)
- Více úroňový partitioning
 - Podle času, podle pobočky
- Možnost individuálního řízení partition
- Omezený počet partition podle implementace

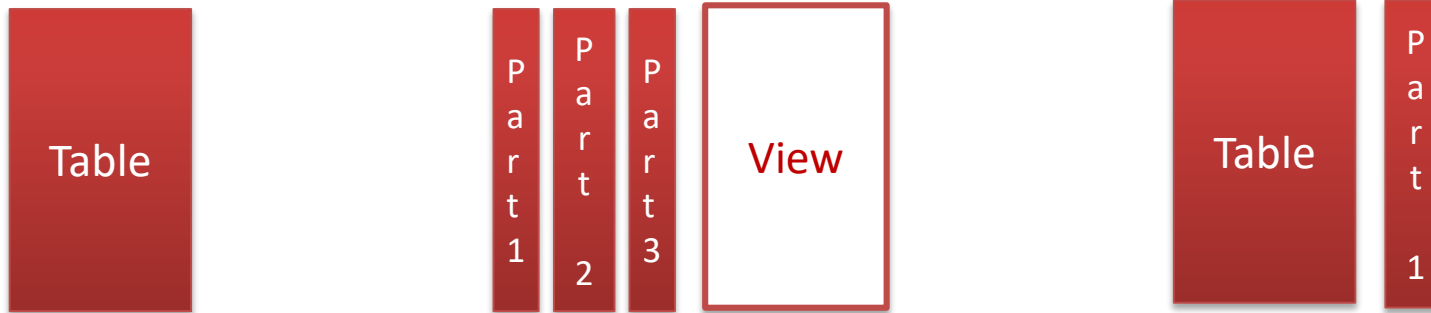


Vertikální rozdělení tabulek

- Přístupy pouze na některé sloupce tabulky
- Příklady:
 - Bloby, obrázky, popisy
- Možnosti
 - Rozdělení tabulek
 - Přidání tabulky (duplicitní záznamy)
 - Vytvoření indexu
- Synchronizace
 - Triggery
 - Aplikační logika

Rozdělení tabulky

Rozdělení tabulky



Výhody

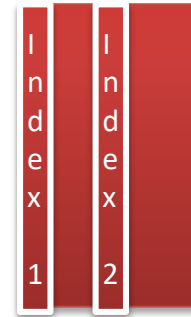
- práce s menším množstvím dat
- méně problémů se zamykáním
- možnost optimalizace

Nevýhody

- nutnost synchronizace – trigger, aplikační logika
- náročnější údržba

Přidání indexů

- Transparentní z pohledu aplikace
- Jeden clustrovaný index
- Libovolný počet dalších indexů
- Automatická údržba
- Pokrývající dotazy
- Nároky na diskový prostor
- Snížení výkonu pro OLTP aplikace



Uložení vypočtených dat

- Přidání sloupce
- Přidání tabulky
- Synchronizace
 - Trigerry
 - Uložené procedury
 - Aplikační logika
- Nutno zavést procedury pro údržbu a resynchronizaci

Materializovaná view

- Transparentní z pohledu aplikace
- Automatické řízení výpočtu view
- Nákladné výpočty, nutnost možnosti řízení výpočtů asynchronně
- Duplicitní uložení dat
 - nároky na diskový prostor

Eliminace nákladných joinů

- Neustálé dotahování hodnot z číselníků
- Omezení datových serverů (Sybase třicet tabulek v jednom joinu)
- Suptype/supertype vazba
- Možnosti
 - Redundantní data
 - Spojení tabulek

Fyzické uložení tabulek

- Cíl
 - Distribuce zátěže na co nejvíce fyzických disků
- Rozložení tabulek a indexů na různé disky
- Podporováno databázemi – definice tablespace, segmentů, datových souborů, ...
- Možnost využití hardware (SAN, NAS, diskové pole)
- Pouze RAID 1+0, dopočítávání kontrolních disků je stále náročné
- Vazba na počet procesorů a procesů – maximalizace paralelního zpracování
- Podpora paralelního zpracování dotazů na straně datového serveru

Co si zapamatovat

- Jaké jsou hlavní kroky při návrhu datového modelu
- Co to je konceptuální model a co obsahuje
- Co je cílem sběru byznys požadavků při vytváření konceptuálního modelu
- Co je logický model a co obsahuje
- Jaké aktivity je nutné provést při převodu konceptuálního datového modelu na logický datový model
- Jaké požadavky je potřeba brát v úvahu při výběru primárních klíčů
- Jaké jsou hlavní rozdíly mezi relačním a objektově orientovaným modelováním
- Kdy a proč se vytváří fyzický datový model
- Které aktivity je nutné provést při převodu logického datového modelu na fyzický datový model
- Co to je denormalizace
- Jaké typy denormalizace znáte

```

4780 GOTO 5000
4790 :
4800 REM -----
4801 REM --- DARSTELLUNG ---
4802 REM --- DES MANUALS ---
4803 REM -----
4810 :
4820 PRINT"00";
4825 W=V+1:IF W<0 THEN W=W+14
4830 FOR X=1 TO 2:PRINT"000000000000";
4835 FOR I=0 TO 23
4840 PRINT MD$(I+W);
4850 NEXT:PRINT:NEXT
4860 PRINT"000000000000";
4870 FOR I=0 TO 23
4880 IF MD$(I+W)=CHR$(32) THEN PRINT MB$
(I+1);:GOTO 4900
4890 PRINT MD$(I+W);
4900 NEXT
4910 PRINT:PRINT"000000000000";
4920 FOR I=2 TO 24 STEP 2
4925 PRINT"|";
4930 IF MD$(I+W-1)="00 00" THEN PRINT"0
0";:GOTO 4940
4935 PRINT" ";
4940 NEXT:PRINT"0"
4950 PRINT"000000000000";
4960 FOR I=2 TO 24 STEP 2
4965 PRINT"|";
4970 IF MD$(I+W-1)="00 00" THEN PRINT"0
0"
MB$(I)"0";:GOTO 4980
4975 PRINT MB$(I);
4980 NEXT:PRINT"0"

```

Diskuse

- Otázky
- Poznámky
- Komentáře
- Připomínky

