INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

VYSOKÁ ŠKOLA BÁŇSKÁ – TECHNICKÁ UNIVERZITA OSTRAVA
FAKULTA STROJNÍ

INFORMAČNÍ SYSTÉMY

Databáze a informační systémy

Ing. Roman Danel, Ph.D.

Ostrava 2013

ISBN 978-80-248-3051-3

Tento studijní materiál vznikl za finanční podpory Evropského sociálního fondu (ESF) a rozpočtu České republiky v rámci řešení projektu: CZ.1.07/2.2.00/15.0463, MODERNIZACE VÝUKOVÝCH MATERIÁLŮ A DIDAKTICKÝCH METOD

OBSAH

1 DATABÁZE A INFORMAČNÍ SYSTÉMY

OBSAH KAPITOLY:

MOTIVACE:

V této kapitole se budeme věnovat databázím, které tvoří základní stavební kámen informačních systémů. Připomeneme si nejvíce používané typy databází a základy relačního databázového modelu, normalizaci databází, dotazovací jazyk SQL a vysvětlení pojmu transakce.

CÍL:

Po prostudování budete rozumět, jaká je souvislost mezi databázemi a informačními systémy. Budete rozumět pojmu transakce, se kterým se u informačních systémů můžete setkat poměrně často. Budete znát, jaké typy databázových systémů existují a k čemu je v databázích využíván jazyk SQL.

1.1 DATABÁZE A INFORMAČNÍ SYSTÉMY

Každý informační systém pracuje s informacemi (a tedy s daty).

Data musí být někde uložena – pro uložení dat se používají **databáze**.

Téměř každý informační systém tedy nějakým způsobem používá databázi (forma může být různá, jako databáze mohou být použity i soubory…).

Základem úspěšného fungování informačního systému je tedy i správné vytvoření databáze a její optimalizace.

V rámci životního cyklu informačního systému, v jeho úvodní fázi, kdy je systém vytvářen, je také vytvářen návrh databáze (volba typu, návrh její struktury, způsob práce, řešení efektivity…).

Audio 5.1 Databáze a informační systémy

1.1.1 Typy databázových systémů

- Manuální systémy - kartotéka
- Agendové systémy – sada programů řeší, obvykle dávkově, určitou „agendu“; data se ručně vpisovala do formulářů, z formulářů se převáděla do el. podoby (děrná páska, děrný štítek)-dávkové zpracování-výstup; závislost na fyzickém uložení dat (struktuře), nízká efektivnost, problémy s redundancí, konzistenci, integritou, dosažitelností, data jsou izolovaná, roztroušená
- Hierarchický model – data v stromové struktuře; rodič-potomek. Příkladem hierarchické databáze je například souborový systém.
- Síťový model – zobecnění hierarchického o mnohonásobné vztahy
- Relační model – E. F. Codd 1970 – data jsou ukládána v tabulkách, mezi kterými jsou definovány vazby; založen na matematickém aparátu množin a predikátové logiky, kolekce tabulek, jejich funkčních vztahů, indexů apod. tvoří databázi; důraz na integritu dat
- Objektově orientovaný model (Gemstone, Caché, ObjektStorem, O2, Versant, …) – data jsou uloženy v objektech spolu s funkcemi pro manipulaci s daty
- Objektově – relační (Oracle)
- „NoSQL“ neboli bezeschémové databáze (CouchDB, MongoDB, Cassandra, Redis, …) – nemají relační strukturu, zpracování semistrukturovaných dat, klíč – hodnota, vhodné pro velký objem dat, využití XML či JSON

Poznámka:
Síťový datový model + objektové typy dat + polymorfismus = objektově-orientovaný model.
Základní návrh databáze vzniká v analytické fázi návrhu informačního systému.
Analýzou rozumíme studium problému před zahájením řešení.

1.1.2 Relační datový model

Úrovně z hlediska abstrakce:

- **Konceptuální model** – rozpoznání základních datových objektů a jejich vztahů – návrh – co je obsahem systému

- **Logický** (technologický) **model** – relační schéma (včetně integritních omezení) – určuje jakou mají data strukturu, není zatížen konkrétní implementací. Často vyjádřen formou ERD a DFD diagramů.
- **Fyzický datový model** – implementace v konkrétním databázovém produktu

1.1.3 Návrh databázového systému

Základní modely vedoucí k návrhu databáze jsou:

- Datový model popisující strukturu, vazby a datové typy (ERD)
- Funkční model – datové a funkční toky (DFD)
- Časová (dynamická) analýza (STD – State Diagram)
- Datový slovník (DD – Data Dictionary) – spojení jednotlivých modelů, odstranění redundance

1.1.3.1 ERD (Entity Relationship Diagram)

ER diagramy slouží k modelování struktury databáze (v grafické podobě).

Cílem ERD je zmapovat data ukládaná do databáze a jejich vzájemné vztahy.

Výstupem ERD je popis logické struktury databáze:

- **Entita** – objekt, který je předmětem zájmu
- **Atribut** – elementární datový prvek, který entitu blíže charakterizuje
- **Relace** – vztah mezi dvěma entitami
- **Kardinalita vztahu** – mocnost vztahu mezi entitami: 1:1, 1:N, N:1, M:N

Vztah je informace, kterou si systém musí pamatovat, nelze ji odvodit.

Pozor, je rozdíl mezi diagramy ERD (Chen) a relačním datovým modelem (RDM, Codd). ERD neuvažuje o tabulkách a řádcích, to už je implementace ERD pomocí RDM.

Tzn. je rozdíl mezi pojmy Relationship (vztah) a Relation (relace)

1.1.3.2 DFD (Data Flow Diagram)

Cílem DFD je modelování datových nebo řídicích toků v systému (v grafické podobě).

DFD diagramy popisují funkce systému.

U DFD diagramů se nejčastěji používá notace dle DeMarca.

DFD se skládá z prvků: „proces", „datový sklad" a „terminátor".

DFD diagramy mají hierarchickou úroveň. Na nejvyšší úrovni stojí **kontextový diagram** – jediný proces který reprezentuje celý systém a naznačuje vztah systému s okolím. Na dalších úrovních jsou popisovány jednotlivé procesy a datové nebo řídicí toky.

Dekompozice DFD na nižší úrovně až na úroveň základních elementárních funkcí:

- Nesmí existovat proces, který nemá žádné vstupy a přesto produkuje datové toky.
- Nesmí existovat proces, který pouze spotřebovává data a nemá výstup.

1.1.3.3 STD (State Transition Diagram)

STD obsahuje sled stavů, v jakém se systém může nacházet a za jakých podmínek může dojít ke změně stavu. Modeluje tedy změny systému v čase.

- Důležitý z hlediska pochopení logiky systému!
- Stavy jsou statické, změna stavu je většinou důsledek nějaké události.
- Vyjádření – např. vývojové diagramy
- Může být hierarchický.

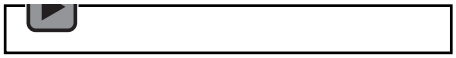
1.1.3.4 DD (Data Dictionary)

Datový slovník slouží k formalizovanému popisu dat systému z pohledu uživatele a odstranění redundance (opakovaného výskytu shodných informací)

Metadata (data o datech):
- Spravuje databázový stroj
- Přístup jen ke čtení

Metadata mohou být například v podobě tabulky, ve které jsou uloženy definice datových tabulek (názvy sloupců tabulky, jejich datový typ, velikost.

Audio 5.2 DD (Data Dictionary)



1.2 SQL

Standardem u dnešních relačních databázových systémů je implementace jazyka SQL.

SQL = Structured Query Language

Deklarativní jazyk, jehož primárním účelem je umožnit práci s daty v relační databázi.

Poznámka:
Pojem „deklarativní jazyk“ znamená, že programátor deklaruje, co chce udělat a ne jak to má být uděláno. Postup vykonání pak řeší příšlušný interpreter – v případě SQL databází to je plánovačdotazů.

SQL příkazy lze rozdělit do dvou skupin:
- **DDL (Data Definition Language)** – vytváření/rušení/modifikace objektů v databázi
- **DML (Data Manipulation Language)** – manipulace s daty

DDL příkazy jsou CREATE, DROP, ALTER, GRANT.

DML příkazy jsou SELECT, INSERT, UPDATE, DELETE.

SQL podléhá standardizaci, jednotlivé normy pro jazyk SQL jsou:
- SQL86
- SQL89 – přidána referenční integrita
- SQL92
- SQL99 – rozšíření o OOP

1.2.1 Množiny

Relační databáze vycházejí z teorie množin. Každá tabulka v databázi je z matematického pohledu množina. Každý řádek tabulky je člen dané množiny.

SQL příkazy jsou tedy vlastně operace nad množinami.

1.2.2 Množinové operace

- Sjednocení - zkombinování (UNION ALL)
- Průnik – nalezení společných prvků (INTERSECT) – základ v Eulerových nebo Vennonových diagramech
- Množinový rozdíl (OUTER JOIN) – prvky, které jsou v jedné a nejsou v druhé množině
- Symetrický rozdíl
- Kartézský součin
- Projekce, Restrikce

1.2.3 Normalizace

Cílem normalizace je dosažení ideální struktury dat. Tj. návrh správných relací bez nadbytečných informací.

Normalizace se snaží vytvořit relační model pro uložení dat, který minimalizuje datovou redundanci (vícenásobné uložení stejných dat) při zachování datové integrity (konzistence).

Existují tzv. „normální formy“ (NF) databáze:

1 NF – záznam neobsahuje žádnou opakující se položku

2 NF – záznam má pouze primární klíč (PK) a neklíčové položky jsou závislé na celém PK

3 NF – vylučuje tranzitivní (přenesené) závislosti – do tabulky nepatří záznamy, které nejsou součástí klíče (tj. obsahem sloupce jsou jednoduché, dále nedělitelné informace)

BCNF – Boyce-Codd Normal Form = 4 NF

Existuje i čtvrtá a pátá NF, v praxi se ale příliš nepoužívají, nejčastěji se návrh optimalizuje do úrovně 3NF.

Špatný návrh relačního modelu vede k následujícím důsledkům:

- Redundance dat
- Nemožnost vyjádřit určité informace

1.2.4 Denormalizace

- Ne vždy je ideální maximální normalizace – důsledná normalizace může snižovat rychlost zpracování
- Denormalizace – agregovaná data, předpřipravené hodnoty pro sestavy – zrychlují výstupy

1.2.5 Integrita

- **Entitní**
  PK (Primary Key, primární klíč) – jednoznačná identifikace entity (každá hodnota PK musí být jednoznačná a také nesmí být hodnota NULL)
- **Referenční**
  FK (Foreign Key, cizí klíč) – hodnota sloupce, která se odkazuje na hodnotu v sloupci jiné tabulky. V tabulce nemůže být záznam s hodnotou, která se nevyskytuje v tabulce, na níž se klíč odkazuje. Pomocí cizího klíče jsou realizovány vazby v relační databázi.

Referenční integrita:

- **Restriktivní**
  Nelze smazat nadřízený záznam, pokud na něj existuje odkaz z nějakého podřízeného záznamu.
- **Set null**
  Pokud smažeme nadřízený záznam, v podřízeném záznamu se hodnota cizího klíče nastaví na NULL.
- **Kaskádová**
  Pokud smažeme nadřízený záznam, smažou se i všechny podřízené záznamy, které se na něho odkazují.

1.2.6 Transakce

Transakce – skupina operací modifikujících data v databázi, které musíme vykonat, abychom dosáhli cíle a přitom databáze zůstala konzistentní. **Transakce je nedělitelná, všechny operace v transakci musí proběhnout jako celek nebo se v případě neúspěchu musí výsledky transakce zrušit (rollback) a vrátit transakcí modifikovaná data do původního stavu.**

Konzistence může být porušena např. při paralelním zpracování více SQL příkazů, které pracují nad stejnými daty.

Vlastnosti transakce

A C I D ***- Atomicity, Consistence, Isolation, Durability***

- **Atomicity** – transakce je atomická, tj. dále nedělitelná, je chápána jako celek
- **Consistence** – výsledkem proběhnuvší transakce musí být data v konzistentním stavu (tzn., musí splňovat všechny integritní a referenční omezení)
- **Isolation** – data a změny zpracovávané transakcí jsou do ukončení transakce pro ostatní uživatele neviditelné
- **Durability** – výsledek transakce musí být trvalý (důsledkem tohoto požadavku je, aby data změněná v důsledku potvrzené transakce byly zapsány fyzicky na disk a ne pouze v cache paměti databázového stroje).

Systémy založené na transakčním zpracování označujeme jako **OLTP** (On-line Transaction Processing). Zkratkou OLTP se označují běžné provozní systémy s relační databází, ve které uživatelé čtou i zapisují data. Alternativou k OLTP jsou **OLAP** systémy (datové sklady), které slouží k analýze dat. Bližší informace k OLAP systémům se nacházejí v následující kapitole.

Důvodem pro zavedení transakcí bylo řešení potíží s víceuživatelským přístupem do databází a snaha zabránit, aby si uživatelé vzájemně nepřepisovali data.

Ukončení transakce

Transakce může být ukončena příkazy **COMMIT** (potvrzení platnosti) nebo **ROLLBACK** (zrušení transakce, návrat zpět).

Audio5.3 Transakce

1.2.7 Vztahy mezi tabulkami v relační databázi – shrnutí

- Databáze obsahuje řadu tabulek
- Všechny tabulky spolu nemusí souviset
- Tabulky, které jsou ve vzájemném vztahu, vytvářejí „schéma“
- Databáze může obsahovat několik schémat

1.2.8 Indexy

Indexy se definují nad sloupcem nebo skupinou sloupců v tabulce a slouží k **zrychlení čtení a vyhledávání dat z tabulky**. Je-li požadováno vyhledání dat v sloupci, který je součástí indexu, nemusí databázový stroj načítat celou tabulku (sekvenční scan), ale načítá pouze datové stránky podle informací uložených v indexu, což vede k rychlejšímu průběhu dotazu.

1.3 ROZHRANÍ PRO PŘÍSTUP NA DATA

Na data uložená v databázovém systému přistupujeme přes rozhraní. Kromě nativních rozhraní poskytovaných databázovými systémy, můžeme ještě využít některé z univerzálnějších API rozhraní, jako jsou např.

ODBC	Open Database Connectivity
JDBC	Java Database Connectivity
ADO	ActiveX Data Objects (Microsoft)

1.4 DATOVÉ MODELOVÁNÍ

Datové modelování je specifická část softwarového inženýrství (není to programování, ale ani pouhé kreslení diagramů), cílem datového modelování je převod reálných objektů na datové objekty (struktury).

1.4.1 Teoretické základy

- Teorie množin
- Turingův stroj – nejedná se o fyzický stroj, ale matematickou konstrukci založenou na manipulaci s buňkami v paměti a používání instrukcí -> teoretický základ konstrukce počítače, založeném na myšlence univerzálního stroje, který pomocí omezené sady jednoduchých instrukcí je schopen řešit jakýkoli algoritmus
- Kleen, Hilbert – rekurzivní funkce -> konstrukce algoritmů, programovací jazyky
- Lambda-kalkul (A. Church) – nástroj k zápisu a manipulaci s počítačovým kódem (tj. s příkazy, které říkají co a jak má počítač dělat) – lambda kalkul nakládá s příkazy stejnou formou jako s matematickými výrazy; nástroj, pomocí kterého lze vyjádřit co má počítač dělat bez nutnosti použít konkrétní programovací jazyk

1.5 OBJEKTOVĚ ORIENTOVANÉ DATOVÉ MODELOVÁNÍ

- Základem je „objekt“ – modeluje nějakou část reálného světa
- Objekty dokážou reagovat na požadavky (zprávy), které jim zasíláme
- Objekty v sobě uchovávají údaje – vnitřní data (Properties) a operace, které lze prostřednictvím zpráv spouštět a provádět (metody)

Dvě koncepce objektově-orientovaného programování (OOP):

- Čisté OOP – ideální pro programování a datové modelování, ale obtížná praktická realizace (programovací jazyk SmallTalk, databáze Gemstone)
- Smíšená (hybridní) – ztrátou některých abstraktních vlastností OOP docílíme zjednodušení implementace (Př. Java, C++, databáze Oracle, Caché...)

Vlastnosti:

- **Objekt** je pro nás „černou skříňkou – zajímá nás, co dělá a ne z čeho se skládá (zapouzdření neboli encapsulation). Objekt má atributy (vlastnosti) a metody (funkce).
- **Dědičnost** (možnost vytvářet nové instance objektů)
- **Polymorfismus** – překrývání metod, které jsme zdědili – metoda stejného jména může mít trochu jinou funkcionalitu, přestože je tato funkcionalita pojmově blízká. Analogie je pojem „otevřít“ a rozdíl mezi funkcí „otevřít dveře“ a „otevřít láhev“
- **Rozhraní (interface)** = skupina metod

1.5.1 Objektově orientovaný a relační model

- Relační model – prvky reálného světa se snažíme zobrazit do pevných předem připravených struktur
- OO model – pro prvky reálného světa vytváříme objekty, které se jim podobají

ERD:	Entita, Atribut, Relace
OO model:	Třída, Objekt, Atribut, Metoda, Vztah
Sdružení ODMG:	www.odmg.org
Další informace:	http://en.wikipedia.org/wiki/Object-oriented_programming

2 KONTROLNÍ OTÁZKY

1. Co je to databáze a jaký je její vztah k informačnímu systému?
2. Jaké znáte druhy databází?
3. Jaké jsou základní charakteristiky relačního datového modelu?
4. Co je to SQL?
5. Co je to normalizace databáze?
6. Co je to transakce a k čemu se používá?
7. Co to jsou OLTP a OLAP systémy?
8. Jaké jsou rozhraní pro přístup na data?

3 POUŽITÁ LITERATURA

[1] Chlapek, D. - Řepa, V. - Stanovská, I.: Vývoj informačních systémů (pracovní sešit ke cvičením). VŠE Praha, 2005. ISBN: 80-245-0977-6.

[2] Hernandez, J. M.: Návrh databází. Grada, 2005. ISBN: 80-247-0900-7.