MUNI SCI

Databázové systémy a R v datové vědě

Veronika Eclerová
eclerova@math.muni.cz

Přírodovědecká fakulta, Masarykova Universita

28. března 2022

Úvod do databázových systémů

- Úvod do datové vědy
- Entity-relationship model
- Relační databáze

Úvod do jazyka SQL

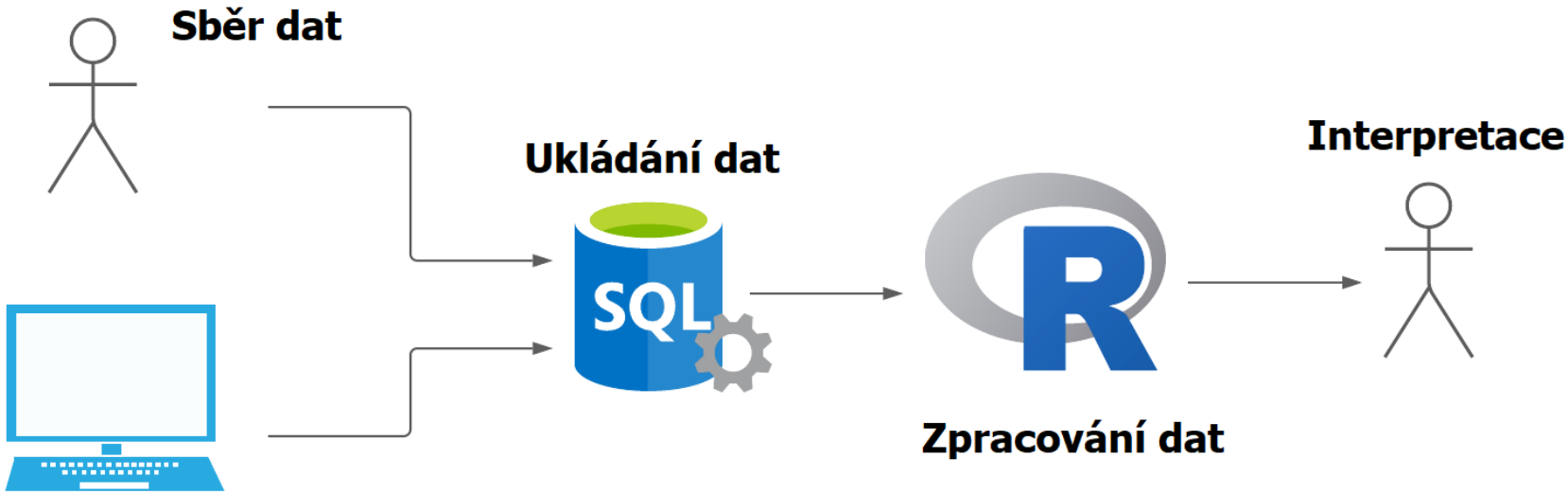
- Základní příkazy jazyka SQL
- DML - pohledy, uložené funkce
- DDL - vytváření tabulek, mazání tabulek, úprava tabulek
- Systémové databáze a jejich funkce
- Programování v SQL (T-SQL)

Datová věda
Data Science

- metody pro sběr dat
- zpracování dat
- interpretace dat

Věda založená na datech
Data intensive science

- astronomie, molekulární biologie, epidemiologie …
- objev jsou podpořeny nasbíranými a zpracovanými daty



Johan Gregor Mendel[1]

1822-1884

- je považován za otce genetiky
- 1866 publikoval *Pokusy s rostlinnými hybridy*

[1] mendelmuseum.muni.cz/o-muzeu/gregor-johann-mendel

Postup při návrhu databáze

- analýza
- tvorba datového modelu
- návrh designu databáze
- implementace

Cíle

1. uklání dat
2. změna dat
3. vyhodnocení dat

Tvorba datového modelu

Entity-relationship model = E-R model I

- entita = konkrétní věc, kterou chce uživatel sledovat (například konkrétní územní celky: okres Benešov, Středočeský kraj)
- třída entit = soubor všech entit (např. všech okresů, krajů)
- atribut = charakteristika dané entity

Speciálními atributy jsou identifikátory.

Jedná se například o atributy příjmení, název, kód (např. CZNUTS). Kód CZNUTS je jednoznačný identifikátor. Naopak příjmení obecně není jednoznačný identifikátor. Jméno a příjmení je složený identifikátor.

Tvorba datového modelu

Entity-relationship model = E-R model II

- vztah (relationship) = vazba entity na jinou entitu

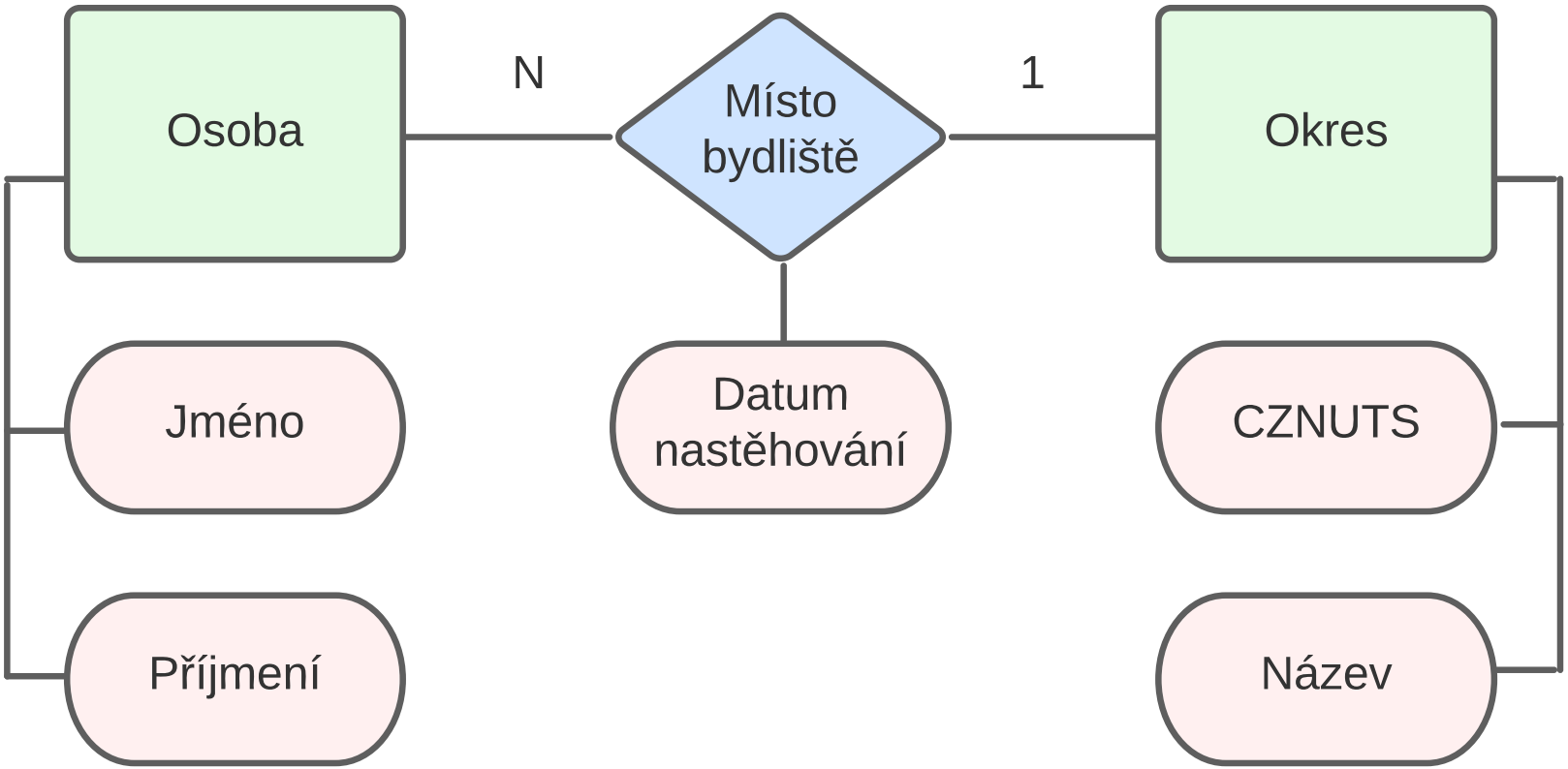
Binární vztah

Vztah mezi dvěma třídami entit. Existuje několik typů binárních vztahů

- 1:1 - one-to-one
- 1:N - one-to-many
- N:M - many-to-many

Speciálním typem vztahů jsou rekurzivní vztahy, které mohou sloužit například k definování stromových struktur.

- ke grafické reprezentaci E-R modelu slouží entity-relationship (E-R) diagramy


Osoba
N
Místo bydliště
1
Okres
Jméno
Datum nastěhování
CZNUTS
Příjmení
Název

Relační databáze

- základním stavebním blokem jsou **datové tabulky** neboli **relace**
- v datové tabulce jsou uložena data týkající se jedné oblasti
- **řádky** datové tabulky = **záznamy**: obsahují data týkající se konrétního případu/výskytu = **instance**
- **sloupce** datové tabulky = **pole**: obsahují konkrétní charakteristiku sledovanou pro všechny řádky
- pokud pro některé záznam není uvedena hodnota některého pole, je v databázi reprezentována **NULL** hodnotou
- záznamy v datové tabulce je možné jednoznačně identifikovat pomocí pole tzv. **primárního klíče**
- **kandidátní klíč** je atribut (skupina atributů), kterým jsou jednoznačně určeny řádky v tabulce

Primární klíč

Často je realizován pomocí sloupce ID - sloupec nenese žádný význam, ale zajišťuje jednoznačnou identifikaci.

Příklad - datová tabulka okres

sloupce

řádky

ok_kod	ok_nazev_kratky	ok_nazev_dlouhy	ok_cznuts	ok_rozloha	ok_pocet_obci	kr_kod
1	Benešov	Benešov	CZ0201	1474.69	114	2
2	Beroun	Beroun	CZ0202	661.91	85	2
3	Blansko	Blansko	CZ0641	862.65	116	11
4	Brno-město	Brno-město	CZ0642	230.22	1	11
5	Brno-venkov	Brno-venkov	CZ0643	1498.95	187	11
6	Bruntál	Bruntál	CZ0801	1536.06	67	14
7	Břeclav	Břeclav	CZ0644	1048.91	63	11
8	Česká Lípa	Česká Lípa	CZ0511	1072.91	57	7
9	Č. Budějovice	České Budějovice	CZ0311	1638.30	109	3
10	Český Krumlov	Český Krumlov	CZ0312	1615.03	46	3
11	Děčín	Děčín	CZ0421	908.58	52	6
12	Domažlice	Domažlice	CZ0321	1123.46	85	4
13	Frýdek-Místek	Frýdek-Místek	CZ0802	1208.49	72	14
14	Havlíčkův Brod	Havlíčkův Brod	CZ0631	1264.95	120	10
15	Hodonín	Hodonín	CZ0645	1099.13	82	11
16	Hradec Králové	Hradec Králové	CZ0521	891.62	104	8
17	Cheb	Cheb	CZ0411	1045.94	40	5
18	Chomutov	Chomutov	CZ0422	935.30	44	6
19	Chrudim	Chrudim	CZ0531	992.62	108	9

- vztahy mezi tabulkami jsou realizovány prostřednictví primárního klíče
- primární klíč jedné tabulky je uložen jako pole druhé tabulky, toto pole se nazývá **cizí klíč**

Primární a cizí klíč

Tabulka *okres* obsahuje pole *kr_kod*, které je cizím klíčem do tabulky *kraj*.

kr_kod	kr_nazev	kr_cznuts	ob_kod
1	Hlavní město Praha	CZ010	1
2	Středočeský kraj	CZ020	2
3	Jihočeský kraj	CZ031	3
4	Plzeňský kraj	CZ032	3
5	Karlovarský kraj	CZ041	4
6	Ústecký kraj	CZ042	4
7	Liberecký kraj	CZ051	5
8	Královéhradecký kraj	CZ052	5
9	Pardubický kraj	CZ053	5
10	Kraj Vysočina	CZ063	6
11	Jihomoravský kraj	CZ064	6
12	Olomoucký kraj	CZ071	7
13	Zlínský kraj	CZ072	7
14	Moravskoslezský kraj	CZ080	8

Základní charakteristiky relačního modelu

1. Řádky obsahují informace o entitách
2. Sloupce obsahují hodnoty atributů vztahující se k dané entitě
3. Každé pole tabulky obsahuje jedinou hodnotu
4. Všechny údaje v jednom sloupci mají stejný typ (domain integrity constraint)
5. Každý sloupec má v rámci tabulky jednoznačné jméno
6. Na pořadí řádků nezáleží
7. Na pořadí sloupců nezáleží
8. Žádné dva řádky neobsahují shodná data

Tabulka nesplňující relační schéma

Kód	Jméno	Příjmení	Oddělení	Email	Komentář
1	Jan	Novák	Kancelář ředitele	novak@spolecnost.cz	Ředitel společnosti. Převzal řízení v roce 1998.
2	Petr	Nový	Lidské zdroje	novy@spolecnost.cz	Zaměstnanec měsíce.
3	Alena	Zelená	Právní oddělení	zelena@spolecnost.cz	Vedoucí oddělení.
				zelena@soukromy_email.cz	
4	Petra	Světlá	Lidské zdroje	svetla@spolecnost.cz	Vedoucí oddělení.

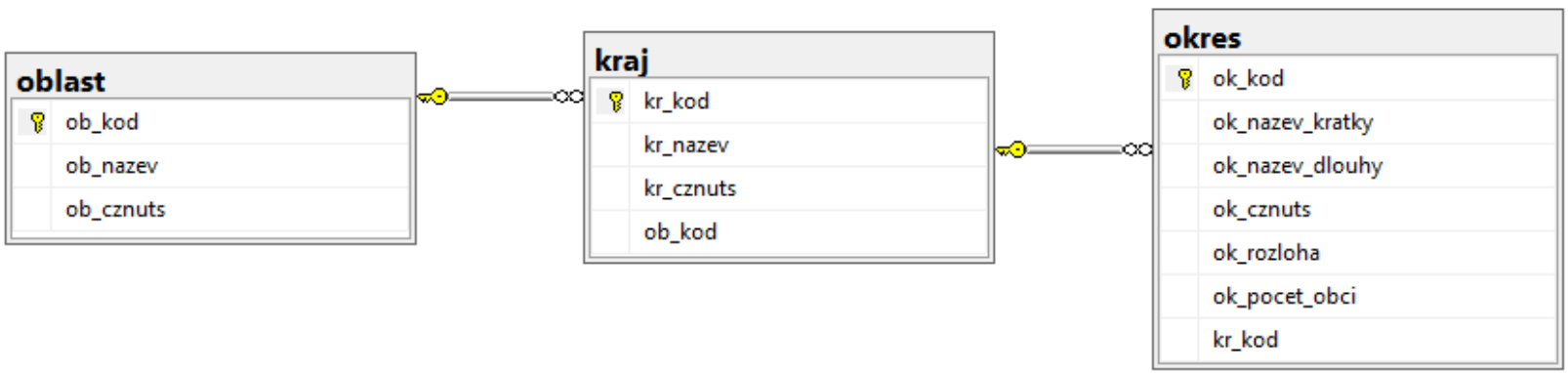
Tabulka nesplňující relační schéma

Kód	Jméno	Příjmení	Oddělení	Email	Komentář
1	Jan	Novák	Kancelář ředitele	novak@spolecnost.cz	Ředitel společnosti. Převzal řízení v roce 1998.
2	Petr	Nový	Lidské zdroje	novy@spolecnost.cz	Zaměstnanec měsíce.
3	Alena	Zelená	Právní oddělení	zelena@spolecnost.cz	Vedoucí oddělení.
				zelena@soukromy_email.cz	
4	Petra	Světlá	Lidské zdroje	svetla@spolecnost.cz	Vedoucí oddělení.

- komentář v řádku s kódem 1 obsahuje více než jednu informaci
- u paní Zelené v řádku 3 jsou uvedeny dva emaily.

Databázový diagram

Grafická reprezentace databáze, která obsahuje názvy jednotlivých datových tabulek a jejich polí. Označuje primární klíče datových tabulek a znázorňuje relace mezi datovými tabulkami.



Příklad - reprezentace binárních vztahů

Uvažujte následující tři situace a rozhodněte, která nejlépe odpovídá binárnímu one-to-one, one-to-many, many-to-many:

- Uvažujte skupinu vysokoškolských studentů, kteří si zapisují různé předměty. U studentů sledujeme atributy: jméno, příjmení, ročník, typ studia. U předmětů sledujeme atributy: název, vyučující, rozvrhovaný čas začátku a konce.
- Uvažujte skupinu tanečnic a tanečníků. U tanečnic sledujeme: jméno, příjmení, tělesné míry a výšku. U tanečníků sledujeme: jméno, příjmení, výšku a váhu. Dále sledujeme informaci o vytvořených tanečních párech.
- Uvažujte skupinu zaměstnanců, kteří jsou zaměstnání na různých odděleních dané firmy. U zaměstnanců sledujeme: jméno, příjmení, pracovní pozici. U oddělení sledujeme: název, jméno ředitele, adresu sídla.

Navrhněte různé způsoby ukládání těchto dat.

Příklad - uložení dat o organizačním členění v České republice

Uvažujme data, která obsahují různé organizační celky v České republice (okresy, kraje, oblasti). U každého s těchto celků známe

- zkrácený název
- název
- kód CZNUTS.

V případě okresu víme, do kterého patří kraje. V případě kraje víme, do které patří oblasti. U okresu známe jeho počet obcí a rozlohu. Navrhněte různé způsoby ukládání těchto dat.

Příklad - uložení dat o hlášených případech COVID 19

Uvažujme data, která obsahují různé hlášení o událostech (úmrtí, vyléčení, uzdravení) v souvislosti s nemocí COVID 19 v České republice. U každé události známe

- datum, kdy nastala
- věk osoby, u které nastala
- pohlaví osoby, u které nastala
- okres, ve kterém nastala.

Navrhněte různé způsoby ukládání těchto dat.

Příklad - uložení dat o historii bydliště

Uvažujme data, která obsahují pro různé osoby informace o jejich bydlišti. Chcete ukládat nejen aktuální data, ale i všechna předešlá bydliště dané osoby, konkrétně:

- jméno a příjmení osoby
- rok, kdy osoba začala bydlet na daném místě
- okres, ve kterém osoba bydlí/bydlela
- zpřesňující informace k adrese (přesná adresa, město ...).

Navrhněte různé způsoby ukládání těchto dat.

Princip návrhu relačních databází, normalizace

- **kandidátní klíč** je atribut (skupina atributů), který jednoznačně identifikuje řádek relace (tabulky)
- **funkční závislost** je vztah mezi dvěma atributy v rámci jedné tabulky
- Atribut Y je funkčně závislý na atributu X, pokud každá platná hodnota X určuje Y jednoznačně. X se nazývá určující atribut (determinant), Y se nazývá závislý atribut (dependent).

Funkční závislost

Atributy jméno, příjmení, věk, pohlaví jsou funkčně závislé na atributu rodné číslo.

Princip návrhu relačních databází

1. Relace (tabulka) se nazývá správně vytvořená (well-formed), jestli je každý její určující atribut zároveň kandidátním klíčem.
2. Každá relace, která není správně vytvořená, by měla být rozdělena na jednu nebo více dobře vytvořených relací.

- **normalizace** proces zkoumání a přetváření relací tak, aby byly správně vytvořeny (dle principu návrhu relačních databází)
- **normální formy** jsou sady pravidel, jejichž splnění vyžadujeme v relačním databázovém návrhu

Proces normalizace

1. Identifikujte všechny kandidátní klíče.
2. Identifikujte všechny funkční závislosti.
3. Zjistěte určující atributy všech funkčních závislostí a ověřte, že jsou kandidátními klíči. Pokud nejsou:
   - 3.1 Vytvořte novou relaci (tabulku), do které umístěte všechny určující i závislé atributy problematické funkční závislosti.
   - 3.2 Jako primární klíč nové relace zvolte určující atribut/y sledované funkční závislosti.
   - 3.3 Ponechte kopii určujícího/ch atributu/ů v původní relaci a označte je jako cizí klíč.
4. Opakujte krok 3.

Příklad - uložení dat o ubytování studentů na kolejích

Uvažujme data o studentech a kolejích, na kterých jsou ubytování. Příklad takto získaných dat je vidět na obrázku:

Student								Koleje/Privát				
Jméno	Příjmení	Věk	Adresa	Okres	Kraj	Ročník	Spolubydlící	Název	Adresa	Okres	Kraj	Cena za pokoj za měsíc
Petr	Novák	21	Nová ulice 1, Opava	Opava	Moravskoslezský	2	Martin Starý	Koleje Kounicova	Kounicova 507/50	Brno	Jihomoravský kraj	15 000
Matin	Starý	22	Moje ulice 2, Zlín	Zlín	Zlínkský kraj	3	Petr Novák	Koleje Kounicova	Kounicova 507/50	Brno	Jihomoravský kraj	15 000
Alena	Bílá	23	Domov 1, Plzeň	Plzeň-město	Plzeňský	3	Sandra Studená	Koleje Komárov	Bří Žůrků 591/5, Brno	Brno	Jihomoravský kraj	12 000
Sandra	Studená	23	Hlavní ulice 3, Ústí nad Labem	Ústí nad Labem	Ústecký	3	Alena Bílá	Koleje Komárov	Bří Žůrků 591/5, Brno	Brno	Jihomoravský kraj	12 000
Pavel	Nový	21	Hlavní ulice 5, Ostrava	Ostrava	Moravskoslezský	1		Koleje Kounicova	Kounicova 507/50	Brno	Jihomoravský kraj	15 000
Alexandra	Novotná	21	Vedlejší ulice 12, Karlovy Vary	Karlovy Vary	Karlovarský	1		Koleje Kounicova	Kounicova 507/50	Brno	Jihomoravský kraj	15 000
Pavlína	Nová	23	Novobranská 2, Pardubice	Pardubice	Pardubický	3	Petra Levá, Alena Pravá	Koleje Komárov	Bří Žůrků 591/5, Brno	Brno	Jihomoravský kraj	12 000
Petra	Levá	23	Hlavní ulice 3, Hodonín	Hodonín	Jihomoravský	3	Pavlína Nová, Alena Pravá	Koleje Komárov	Bří Žůrků 591/5, Brno	Brno	Jihomoravský kraj	12 000
Alena	Pravá	23	Domov 1, České Budějovice	České Budějovice	Jihočeský	3	Pavlína Nová, Petra Levá	Koleje Komárov	Bří Žůrků 591/5, Brno	Brno	Jihomoravský kraj	12 000

Upravte uložení dat tak, aby respektovalo principy návrhu relačních databází.

Hodnocení studentů

Uvažujme data o známkách studentů získaných v jednotlivých předmětech. Příklad takto získaných dat je vidět na obrázku:

Jméno studenta	Příjmení studenta	Předmět	Známka	Datum hodnocení	Jméno vyučujícího	Příjmení vyučujícího
Petr	Novák	Lineární algebra	A	05.05.2021	Petr	Herman
Matin	Starý	Lineární algebra	B	15.05.2021	Petr	Herman
Alena	Bílá	Lineární algebra	B	05.05.2021	Petr	Herman
Sandra	Studená	Lineární algebra	C	15.05.2021	Petr	Herman
Pavel	Nový	Lineární algebra	A	05.05.2021	Petr	Herman
Alexandra	Novotná	Lineární algebra	C	15.05.2021	Petr	Herman
Pavlína	Nová	Lineární algebra	D	05.05.2021	Petr	Herman
Petra	Levá	Lineární algebra	E	15.05.2021	Petr	Herman
Alena	Pravá	Lineární algebra	FE	15.05.2021	Petr	Herman
Petr	Novák	Matematický analýza	B	02.05.2021	Alžeběta	Poslední
Matin	Starý	Matematický analýza	B	02.05.2021	Alžeběta	Poslední
Alena	Bílá	Matematický analýza	FC	02.05.2021	Alžeběta	Poslední
Sandra	Studená	Matematický analýza	C	14.05.2021	Alžeběta	Poslední
Pavel	Nový	Matematický analýza	B	02.05.2021	Alžeběta	Poslední
Alexandra	Novotná	Matematický analýza	C	02.05.2021	Alžeběta	Poslední
Pavlína	Nová	Matematický analýza	D	02.05.2021	Alžeběta	Poslední
Petra	Levá	Matematický analýza	D	02.05.2021	Alžeběta	Poslední
Alena	Pravá	Matematický analýza	FFC	20.05.2021	Alžeběta	Poslední
Petr	Novák	Statistika	A	01.06.2021	Alexandr	Trojan
Matin	Starý	Statistika	B	01.06.2021	Alexandr	Trojan
Alena	Bílá	Statistika	C	01.06.2021	Alexandr	Trojan
Sandra	Studená	Statistika	C	01.06.2021	Alexandr	Trojan
Pavel	Nový	Statistika	D	01.06.2021	Alexandr	Trojan
Alexandra	Novotná	Statistika	C	01.06.2021	Alexandr	Trojan
Pavlína	Nová	Statistika	D	01.06.2021	Alexandr	Trojan
Petra	Levá	Statistika	C	01.06.2021	Alexandr	Trojan
Alena	Pravá	Statistika	E	01.06.2021	Alexandr	Trojan

Upravte uložení dat tak, aby respektovalo principy návrhu relačních databází.

Integritní omezení

Pravidla/systémy pravidel, která zaručují integritu databáze.

Příklady

- unikátnost zadaných hodnot v rámci datových tabulek
- daný datový typ nebo rozsah hodnot v tabulkách
- primární a cizí klíče

Hodnocení studentů

Uvažujme data o známkách studentů získaných v jednotlivých předmětech (viz předchozí příklad). Vymyslete vhodná integritní omezení pro tato data.

Základy jazyka SQL

SQL = Structured Query Language

Základní součásti jazyka SQL:

- **Data definition language (DDL)**: vytváření tabulek, vztahů a dalších struktur datbáze
- **Data manipulation language (DML)**: pohledy na data, ukládání, změna a mazání dat
- **SQL/Persistent stored modules (SQL/PSM)**: příkazy, které umožňují procedurální programování v SQL
- **Transaction control language (TCL)**: příkazy, které umožňují definování transakcí a jejich řízení
- **Data control language (DCL)**: příkazy umožňují zprávu přístupu k databázi

DML - pohledy na data

Základní syntaxe

select	výběr polí tabulky
from	výběr tabulky
where	podmínky na zobrazované záznamy

DATOVÉ TYPY

- řetězec: char, varchar
- číslo: int, numeric, float
- datum: datetime
- identifikátory: uniqueidentifier

Klíčové slovo **AS** slouží k přejmenování sloupců tabulky/celých tabulek v rámci dotazů. Je vhodné při použití klausule SELECT a při spojování tabulek.

Příklad - výpis dat z tabulky okresů

Vypište dlouhý název a CZNUTS okresů za následujících podmínek:

- všech okresů ČR
- pouze okresů v Jihomoravském kraji
- pouze okresů v Jihomoravském kraji nebo Zlínském kraji
- pouze okresů, které nejsou v Jihomoravském kraji ani Zlínském kraji
- okresů, kde počet obcí převyšuje 100
- okresů, jejichž rozloha je mezi 700 a 800 km^2

Výpis opakujte, ale tentokrát zobrazte všechny sloupce tabulky.

ORDER BY slouží k seřazení záznamů ve výpisu dat, zapisuje se za klausuli „where“

klíčová slova **ASC** a **DESC** slouží k určení vzestupného resp. sestupného pořadí záznamů (vzestupné řazení je výchozí)

Příklad - výpis dat z tabulky okresů

Vypište dlouhý název a CZNUTS okresů za následujících podmínek:

- všech okresů ČR seřazených abecedně podle dlouhého názvu
- pouze okresů v Jihomoravském kraji seřazených abecedně podle dlouhého názvu
- všech okresů ČR seřazených vzestupně/sestupně podle počtu obcí
- všech okresů ČR seřazených nejdříve podle počtu obcí vzestupně a v případě stejného počtu obcí podle rozlohy sestupně

Rozšířená syntaxe

select	výběr polí tabulky
from	výběr tabulky
where	podmínky aplikované na tabulku před agregací
group by	seznam sloupců, podle kterých se agreguje
having	podmínky na zobrazené záznamy
order by	sloupce, podle kterých se provádí řazení

Příklady agregačních funkcí

MAX, MIN, COUNT, COUNT + DISTINCT, SUM, AVG, VAR
další agregační funkce lze najít na tomto odkazu

Příklad - výpis dat z tabulky o případech nákazy COVID-19

- Vypište celkový počet nově nakažených, uzdravených a vyléčených v celé tabulce.
- Kolik řádků v tabulce nemá vyplněný okres?
- Vypište celkový počet nově nakažených, uzdravených a vyléčených po jednotlivých dnech v dubnu a květnu 2020. Data vhodně seřaďte.
- Vypište celkový počet nově nakažených, uzdravených a vyléčených po jednotlivých měsících. Data vhodně seřaďte.
- Vypište průměrný, maximální a minimální počet nově nakažených mužů za den. Jakou má výpis limitaci?
- Vypište průměrný počet nově nakažených mužů za den po měsících.
- Vypište měsíc, ve kterém byl nejvyšší počet nově nakažených a jejich počet.

Spojování tabulek

- cross join: jedná se o operaci, která zprostředkovává kartézský součin tabulek; syntaxe v jazyce SQL je oddělení jednotlivých tabulek čárkou
- (inner) **JOIN**: je podmnožinou kartézského součinu tabulek; podmnožina je definována logickým výrazem uvedeným za klíčovým slovem **ON**
- **LEFT JOIN**: výsledná tabulka obsahuje (i) všechny záznamy „levé“ tabulky, (ii) podmnožinu kartézského součinu, která je definována logickým výrazem uvedeným za klíčovým slovem **ON**; pokud záznam z „levé“ tabulky nesplňuje podmínky na přiřazení žádného záznamu „pravé“ tabulky je řádek doplněn hodnotami **NULL**
- **RIGHT JOIN**: lze jej definovat podobně jako **LEFT JOIN**
- **FULL JOIN**: lze jej definovat podobně jako **LEFT JOIN** nebo **RIGHT JOIN**; výsledná tabulka obsahuje (i) všechny záznamy „levé“ i „pravé“ tabulky

Příklad - výpis dat z tabulky okresů a krajů

- Vypište tabulku obsahující všechny okresy společně s příslušnými kraji a oblastmi. Užijte různé varianty datových návrhů.
- Vypište všechny okresy v oblasti Severozápad, které mají více než 100 obcí.
- Vypište průměrný počet obcí v okrese pro všechny okresy v oblasti Severozápad.
- Vypište celkovou rozlohu jednotlivých oblastí ČR.

Příklad - výpis dat z tabulky o případech nákazy COVID-19

- Vypište celkový počet nově nakažených mužů po dnech v Jihomoravském kraji.
- Vypište průměrný počet nově nakažených mužů za den po měsících a krajích.
- Vypište dny, kdy nebyl žádný nově nakažený od počátku března 2020 do konce roku 2020.

Pořadí prováděných operací a vliv na rychlost dotazů

U komplikovanějších dotazů je vhodné dotazy optimalizovat za účelem zvýšení rychlosti nebo snížení užitích zdrojů. Jeden z nástrojů, který se dá využít jsou tzv. „execution plans“,více informací **zde**.

Nástroje používané ke zvýšení rychlosti

- indexy - struktura uložená na disku, která umožňuje rychlé vyhledávání v tabulkách a spojování tabulek. Automaticky se vytváří pro primární klíče.
- dočasné tabulky - umožňují dočasně uložit část dat (například agregovaných) potřebných pro běh dotazu.

Příklad - execution plans, indexy

V obou následujících případech užijte příkazy:
`set statistics time on` na začátku skriptu
`set statistics time off` na konci skriptu.
Zkorodujte si `SQL Server Execution Times` na záložce Messages.

- Z tabulky `nakaza` vypište celkový počet případů nákaz zaznamenaných v Jihomoravském kraji. Dotaz proveďte (i) s použitím příkazu `join`, (ii) s použitím vnořeného dotazu. Zobrazte v obou případech předpokládaný a reálný execution plan. Vytvořte vhodný index na tabulce `nakaza`. Výpis opakujte.
- Viz slide 30: Vypište měsíc, ve kterém byl nejvyšší počet nově nakažených a jejich počet. Výpis proveďte bez použití dočasné tabulky a s použitím dočasné tabulky.

Pohledy, uložené funkce

Pohled na data

CREATE VIEW *název tabulky* AS
SELECT *název sloupce 1*, *název sloupce 2*, ...
FROM *název tabulky (tabulka nebo spojení několika tabulek)*
WHERE *podmínky*

Funkce vracející tabulku

CREATE FUNCTION *název funkce*
(
název parametru 1 a datový typ,
název parametru 2 a datový typ,
...) RETURNS TABLE AS
RETURN
(
SELECT ...
)

Pohledy, uložené funkce

Funkce vracející jednu hodnotu

CREATE FUNCTION *název funkce*
(
název parametru 1 a datový typ,
název parametru 2 a datový typ,
...) RETURNS *datový typ výsledku* AS
BEGIN
DECLARE *názvy lokálních proměnných a datový typ*
SELECT ...
RETURN ...
END

Příklad - uložené funkce se skalárním výstupem

- Vytvořte funkci, která pro zadané datum mezi lety 2010 a 2030 vrátí den v týdnu, který v zadaný den byl/bude.
- Vytvořte funkci, která pro dané rozmezí věků a rok vrátí počet úmrtí na COVID-19.
- Vytvořte funkci, která pro zadaný CZNUTS (okresu, kraje nebo oblasti) vrátí název.

Příklad - pohledy na data

- Vytvořte pohled, který vytvoří tabulku obsahující ve sloupcích název oblasti, kraje, okresu. Užijte různé datové návrhy.
- Vytvořte pohled, který vytvoří tabulku obsahující ve sloupcích název oblasti, kraje, rozlohu kraje a počet obcí v kraji. Užijte různé datové návrhy.
- Viz slide 30: Vypište měsíc, ve kterém byl nejvyšší počet nově nakažených a jejich počet. Výpis proveďte s použitím pohledu. Porovnejte s užitím dočasné tabulky z pohledu (i) rychlosti, (ii) využití diskové kapacity.
- Vytvořte pohled, který vrátí počet pracovních a nepracovních dní v každém měsíci po letech za roky 2022-2024.

Příklad - uložené funkce

- Vytvořte pohled, který vrátí počet pracovních a nepracovních dní pro všechny měsíce zvoleného roku.
- Vytvořte funkci, která pro zvolenou organizační jednotku dle CZNUTS (pro libovolnou úroveň klasifikace) vrátí název této jednotky a všech podřízených.

Ukládání dat

INSERT INTO *název tabulky*(*název sloupce 1*, *název sloupce 2*, ...)
VALUES (*hodnota 1*, *hodnota 2*...)

INSERT INTO *název tabulky*(*název sloupce 1*, *název sloupce 2*, ...)
SELECT ...

SELECT *hodnota 1*, *hodnota 2*...
INTO *název tabulky (nová tabulka)*
FROM *název tabulky (tabulka nebo spojení několika tabulek)*
WHERE *podmínky*

Změna dat a mazání dat

UPDATE *název tabulky*
SET *název sloupce 1=hodnota 1*, *název sloupce 2=hodnota 2*
WHERE *podmínky*

DELETE FROM *název tabulky (tabulka nebo spojení několika tabulek)*
WHERE *podmínky*

Vytváření tabulek, mazání tabulek, úprava tabulek

Vytváření tabulky:
CREATE TABLE *název tabulky* (*název sloupce 1*, *název sloupce 2*, ...)

Přidání sloupce:
ALTER TABLE *název tabulky* ADD *název sloupce* [*datový typ*](*délka datového typu*);

Smazání všech záznamů z tabulky:
TRUNCATE TABLE *název tabulky*

Smazání tabulky:
DROP TABLE *název tabulky*

Příklad - data o onemocnění COVID-19

- Vytvořte novou tabulku, do které vložte agregovaná data obsahující celkový počet nově nakažených osob pro každý den od začátku pandemie do posledního data uvedeného v tabulce.
- Na webu najděte údaje pro 5.5.2021 a vložte je ručně do tabulky.
- Záznamy v tabulce aktualizujte. Vypočtěte rozlohu ČR a data přepočtěte na km^2.
- Smažte z tabulky všechny záznamy, které se netýkají roku 2020.
- Přidejte do tabulky nový sloupec určující den v týdnu.
- Naplňte korektně sloupec „den v týdnu“.
- Smažte z tabulky všechny záznamy.
- Vložte do tabulky záznamy, ale nyní data mají obsahovat pouze celkový počet nově nakažených žen nemocí COVID-19 pro každý den. Nevyplňujte sloupec „den v týdnu“.
- Odstraňte celou tabulku.

Programování v SQL (T-SQL)

Větvení:

IF *logický výraz*
BEGIN *Sekvence příkazů* END
ELSE
BEGIN *Sekvence příkazů* END

CASE *vstupní výraz*
WHEN *výraz k porovnání* THEN *výstupní výraz*
[ELSE *výstupní výraz*]
END

CASE
WHEN *logický výraz* THEN *výstupní výraz*
[ELSE *výstupní výraz*]
END

Cykly:

WHILE *logický výraz*
    BEGIN
        Sekvence příkazů
    END

BREAK a CONTINUE - umožňují v kombinaci z příkazem IF předčasně ukončit cyklus WHILE

Kursor - umožňuje spouštět cyklus přes výsledek příkazu SELECT

DECLARE *název proměnné datový typ*
DECLARE *název kurzoru* CURSOR FOR
Příkaz select
OPEN *název kurzoru*
FETCH NEXT FROM *název kurzoru* INTO *název proměnné*
WHILE @@FETCH_STATUS = 0
    BEGIN
        Sekvence příkazů
        FETCH NEXT FROM *název kurzoru* INTO *název proměnné*
    END
CLOSE *název kurzoru*
DEALLOCATE *název kurzoru*

Uložená procedura:
SET NOCOUNT ON - umožňuje pozastavit výpis počtu provedených operací v SQL, je doporučeno ho použít u uložených procedur

CREATE PROCEDURE *název uložené procedury*
(*název proměnné datový typ*)
AS
BEGIN
SET NOCOUNT ON
Sekvence příkazů
END

Příklad - programování v SQL

- Užijte příkaz CASE k získání celkového počet nákaz, úmrtí a vyléčení pro každý den zvoleného měsíce z tabulky `nakaza`. Dotaz proveďte bez použití vnořeného příkazu SELECT a příkazu JOIN.
- Vytvořte kursor přes všechny měsíce roku 2020. Pro každý s těchto měsíců proveďte příkaz z předchozího bodu.
- Vytvořte novou prázdnou tabulku o čtyřech sloupcích (datum, počet nákaz, úmrtí a vyléčení). Vytvořte uloženou proceduru, která tuto funkci naplní pro zvolený měsíc pomocí dotazu z prvního bodu v této úloze.
- Užijte data z tabulky vytvořené v předchozím bodu a vypočtěte kumulativní počet nákaz, úmrtí a smrtí pro všechny dny v březnu 2021. Užijte příkaz JOIN.

Příklad - programování v SQL

- *Nepovinné:* Vytvořte uloženou funkci, která bude pro zvolený časový úsek (vstupem je datum od a datum do) vypisovat (i) celkový počet nových nákaz, úmrtí a vyléčení, které nastaly před vybraným měsícem (1 řádek); (ii) celkový počet nových nákaz, úmrtí a vyléčení pro každý den zvoleného měsíce; (iii) celkový počet aktivních případů nákazy pro každý den zvoleného měsíce. Data organizujte do jedné datové tabulky. Užijte příkaz UNION.

Trigger

Trigger je databázový objekt, který zahajuje svoji činnost ve chvíli, kdy se v databázi stane nějaká událost.
Triggery dělíme na

- DDL Trigger - spouští se po zavolání příkazu CREATE, DROP nebo ALTER
- DML Trigger spouští se po zavolání (AFTER/FOR Triggery) nebo místo (INSTEAD Of Triggery) příkazu INSERT, UPDATE, DELETE

Systémové databáze a jejich funkce

Katalog objektů systémové databáze

Příklad

Vypište všechny tabulky a sloupce tabulek, které byly vytvořeny v březnu.

```
select o.name,c.name
from sys.columns c join
        sys.objects o on o.object_id=c.object_id
where o.type_desc='USER_TABLE' and month(o.create_date)=3
order by o.object_id, c.column_id
```

MASARYKOVA
UNIVERZITA