Databáze standardu SQL, díl 2.

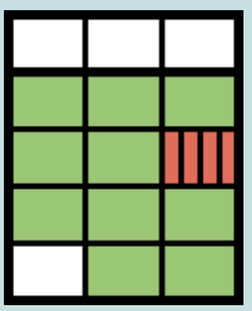
Normální tabulky

Budeme-li chtít v databázovém prostředí řešit reálné úlohy s využitím jazyka SQL, nevyhneme se zavedení základních pojmů Coddova relačního databázového modelu. Tento model vychází z popisu databázového systému pomocí tabulek.

Tabulky v databázovém systému

Tabulka je tvořena jednotlivými řádky a sloupci, které se protínají, a v takto vzniklých políčkách jsou někdy uloženy důležité informace. Tabulku chápejme jako souhrn sloupců, které popisují jednotlivé vlastnosti položek. Z druhého pohledu jde o souhrn řádků, které popisují jednotlivé případy nebo jednotlivé položky. Průsečík řádku a sloupce tabulky nazýváme polem a to je určeno pro uložení uchovávaných hodnot. Je dost důležité uvědomit si, že v databázovém prostředí řádek popisuje přesně jednu jedinou položku, se kterou můžeme samostatně pracovat. Jsou-li například v tabulce ukládány osobní údaje lidí, pak co řádek, to jeden konkrétní člověk. Naproti tomu sloupec je používán k uchovávání informací o jednotlivých vlastnostech položek. V našem případě sloupce v tabulce lidí popisují např. jméno, příjmení, datum narození a podobně. Budete-li studovat odbornou literaturu o databázích, setkáte se s pojmem relace a atribut. Tyto pojmy jsou jen jiným označením pojmů před chvílí zavedených. Atribut není nic jiného než jeden sloupec tabulky popisující jednu vlastnost, zatímco relace je to samé co tabulka, tvořená jednotlivými atributy. Tyto pojmy nepřinášejí nové možnosti. Mohou nanejvýš odradit čtenáře od dalšího čtení tohoto seriálu. Proto v textu nebudou používány. Tabulky je vhodné pro potřeby tohoto článku vhodným způsobem graficky znázornit.

Na obrázku č. 1 je typická ukázka tabulky tak, jak by neměla nikdy vypadat. Vidíme, že tabulka je tvořena jedním řádkem záhlaví a čtyřmi řádky popisujícími čtyři různé položky. Tabulka má tři sloupce určené pro tři různé vlastnosti položek. V následujícím výkladu bude použito pro názornost barevné označení jednotlivých polí. Pole vybarvená zelenou barvou obsahují právě jednu hodnotu. Bílá pole neobsahují žádnou hodnotu. Pole vybarvená červenou barvou obsahují více hodnot. Jak už bylo řečeno, obr. č. 1 je odstrašujícím příkladem, jak by databázová tabulka neměla nikdy vypadat. Takovýmto tabulkám bychom se měli vyhnout, jinak bude v průběhu řešení docházet k velkým těžkostem.



Obr. č. 1. Největší zlo.

Obr. č. 2. Bez klíče.

Na obrázku č. 2 je ukázka jiné tabulky. Tady jde o tabulku, která má nepoměrně lepší vlastnosti než tabulka na obrázku č. 1. Zde už není žádné víceznačné pole a všechna pole jsou buď naplněna jednou hodnotou, nebo v nich hodnota chybí. Cílem druhého dílu seriálu je naučit čtenáře navrhovat tabulky tak, aby byly pokud možno co nejdokonalejší, s využitím nejobecnějších principů jejich dekompozice.

Evidence

Jazyk SQL bývá tradičně vykládán na knihovních systémech. Rozhodl jsem se natruc apelovat na jiné základní znalosti čtenářů, spíše z let předškolních. Myslím, že pohádka O Sněhurce a sedmi trpaslících je hodnotou, na kterou je možno se také odvolávat. Představme si situaci po příchodu Sněhurky k sedmi trpaslíkům a velký zmatek v její duši, když zjistila, že trpaslíci se dramaticky liší, a to zejména povahovými rysy. V tabulce č. 1 vidíme její první pokus o evidenci vlastností trpaslíků.

První sloupec tabulky TRP nese název TRPASLIK a jsou v něm uložena jména čtyř trpaslíků, u nichž už Sněhurka něco zjistila. Ve druhém sloupci s názvem ROKNAR jsou uvedena domnělá data narození trpaslíků (z Brumly se nepodařilo tento údaj vypáčit). Ve třetím sloupci VLASTNOST jsou uvedeny základní vlastnosti jednotlivých trpaslíků. S touto tabulkou bychom mohli být celkem spokojeni, nebýt Šmudly, který má více než jednu vlastnost.

TRPASLIK	ROKNAR	VLASTNOST
SMUDLA	1830	SPINAVY, HODNY, ZAVISLY
PROFA	1807	UCENY
BRUMLA		ZADUMCIVY
KEJCHAL	1857	HODNY

Tab. č. 1. Počátek evidence.

Toto nebezpečí však hrozí u všech trpaslíků, protože zjišťování jejich dalších vlastností v průběhu soužití se Sněhurkou patrně povede k víceznačnosti i u jiných. Pokud by se Sněhurka zastavila v této fázi návrhu, patrně by nikdy nepronikla do tajů návrhu dokonalých tabulek ve vyšších normálních formách. Právě se jí totiž povedlo navrhnout *tabulku v nulté normální formě*. Později se rozhodla odstranit víceznačnosti hodnot. V tabulce č. 2 vidíme nový návrh.

Jde o databázovou tabulku TRPEX. První dva sloupce TRPASLIK a ROKNAR už známe z tabulky č. 1. Byly zavedeny další sloupce, popisující jednotlivé vlastnosti trpaslíků. Tedy zda je trpaslík špinavý, hodný, závislý, učený nebo zádumčivý. V příslušných polích se rozhodla použít logické proměnné typu ANO/NE. Všimněte si, že tabulka č. 2 už má nepoměrně lepší vlastnosti, protože jsou vyloučeny víceznačnosti jednotlivých polí.

V každém poli může nebo nemusí být uvedena hodnota. Pokud by si Sněhurka nebyla zcela jista tím, zda je Šmudla hodný, uvedla by místo YES nebo NO prázdnou hodnotu NULL. Tabulka č. 2 se Sněhurce líbila s jedinou námitkou – spíše odpovídá některým zlozvykům ze spreadsheetu a je málo databázová. Největším argumentem bylo, že pokud by se zjistilo, že je ještě jedna povahová vlastnost, která zatím není uvedena jako sloupec

TRPASLIK	ROKNAR	SPINAVY	HODNY	ZAVISLY	UCENY	ZADUMCIVY
SMUDLA	1830	YES	YES	YES	NO	NO
PROFA	1807	NO	NO	NO	YES	NO
BRUMLA		NO	NO	NO	NO	YES
KEJCHAL	1857	NO	YES	NO	NO	NO

Tab. č. 2. Nové rozhodnutí.

tabulky, ale trpaslíci ji mají, bylo by nutné tabulku rozšířit. Sněhurka navrhla tabulku č. 3.

V této tabulce TRAX se rozhodla každého trpaslíka uvést tolikrát, kolik má vlastností. Proto je zde Šmudla třikrát, Prófa jednou

TRPASLIK	ROKNAR	VLASTNOST
SMUDLA	1830	SPINAVY
SMUDLA	1830	HODNY
SMUDLA	1830	ZAVISLY
PROFA	1807	UCENY
BRUMLA		ZADUMCIVY
KEJCHAL	1857	HODNY

Tab. č. 3. Začínáme přemýšlet.

a Brumla s Kejchalem rovněž. Taková tabulka je sice plýtváním řádků, ale na druhé straně přináší krásnou možnost realizace pouze pomocí tří sloupců s názvy TRPASLIK, ROKNAR a VLASTNOST. Jedinou nepříjemností tabulky č. 3 je zbytečné opakování názvů trpaslíků a roků narození. Jinak lze proti tabulce č. 3 jen těžko něco namítat.

Nultá a první normální forma

Často hovoříme o normálních formách tabulek. Zavedení normálních forem tabulek je neocenitelnou výhodou pro dokonalý návrh databázového systému. Pokud je tabulka v nulté nebo v první normální formě, není ještě žádným velkým zázrakem. Chtěl bych uvést dvě definice:

*Tabulka je v **nulté normální formě** 0NF právě tehdy, když existuje alespoň jedno pole, které obsahuje více než jednu hodnotu.*

*Pokud tabulka není v nulté normální formě, pak je alespoň v **první normální formě** 1NF.*

Důležité je upozornit, že druhá definice neznamená, že tabulka, která není v 0NF, je pouze v 1NF. Tabulka se může dostat i do vyšších normálních forem. Záleží na kvalitě našeho návrhu. Pokud se podíváme na tabulku č. 1 s odstupem, zjistíme, že je v nulté normální formě, protože vlastnost Šmudly je trojznačná. Tabulka č. 2 v nulté normální formě není a bude přinejmenším v první normální formě. (Skutečnost je taková, že tabulka č. 2 je dokonce v páté normální formě.) Tabulka č. 3 také neobsahuje víceznačné hodnoty, a proto není v nulté normální formě. V tomto případě je pouze v první normální formě. Je to způsobeno její nedokonalostí danou opakováním roku narození v jednotlivých řádcích. Jestliže se vrátíme k obrázku č. 1, je na něm znázorněna graficky tabulka v nulté normální formě. Na obrázku č. 2 je znázorněna tabulka, která je pouze v první normální formě, protože nemá některé další vlastnosti související s klíčem. Čím vyšší je normální forma tabulky, tím větší má tato tabulka užitné vlastnosti. Tabulka v nulté normální formě neumožňuje efektivně klást dotazy na její obsah. Naproti tomu je-li tabulka alespoň v první normální formě, je možné klást efektivní dotazy na v ní obsažená data. Pokud se nám podaří dostat tabulku do druhé nebo třetí normální formy, bude to mít pro nás další příjemné následky a usnadní se aktualizace dat.

První SQL dotazy

Někteří čtenáři marně čekali ukázky SQL dotazů v prvním díle tohoto seriálu – ty jsem už zklamal. Ostatní čtenáři, kteří pochopili, že na SQL je zatím dost času, se teď patrně diví, proč předbíhám událostem. Důvod je prostý. Při této příležitosti nechci budovat systematiku těchto dotazů, ale pouze dokumentovat, jak je možné formulovat dotazy do tabulek v nulté a první normální formě, pokud se ovšem neptáme na víceznačné hodnoty. Do tabulky č. 1 v 0NF můžeme klást některé dotazy efektivně. Pokud bychom se chtěli zeptat, ve kterém roce se narodil Prófa, stačí napsat SQL dotaz:

```
SELECT ROKNAR FROM TRP WHERE
TRPASLIK='PROFA';
```

Kdybychom se zajímali o to, kteří trpaslíci se narodili po roce 1820, zvolili bychom jiný dotaz:

```
SELECT TRPASLIK FROM TRP WHERE
ROKNAR>1820;
```

Pak by odpověď obsahovala pouze Šmudlu a Kejchala. Prófa je totiž poněkud starší a u Brumly se zatím neví, kdy se narodil. Z této tabulky můžeme zjistit též Šmudlovy vlastnosti dotazem:

```
SELECT VLASTNOST FROM TRP WHERE
TRPASLIK='SMUDLA';
```

(Bohužel, opačný dotaz na to, kteří trpaslíci mají vlastnost „hodný“, v této tabulce není rozumným způsobem proveditelný.)

To tabulka č. 2 je v páté normální formě, a proto se můžeme klidně zeptat na to, co potřebujeme. Pokud bychom se zajímali o data narození trpaslíků, můžeme klást dotazy formulované třeba takto:

```
SELECT TRPASLIK FROM TRPEX WHERE
ROKNAR=1830;
```

nebo

```
SELECT ROKNAR FROM TRPEX WHERE
TRPASLIK='KEJCHAL';
```

Navíc máme možnost se zeptat, kteří trpaslíci jsou současně hodní a závislí. To učiníme následujícím dotazem:

```
SELECT TRPASLIK FROM TRPEX WHERE
HODNY=YES AND ZAVISLY=YES;
```

Takový dotaz můžeme ještě zjednodušit na tvar:

```
SELECT TRPASLIK FROM TRPEX WHERE HODNY
AND ZAVISLY;
```

To si můžeme dovolit, neboť logické hodnoty se dají přímo použít ve výrazech. Takovou možnost jsme v tabulce č. 1 neměli. Je velkou výhodou, že se můžeme u lépe navržené tabulky dobře zeptat. Pokud si ovšem někdo myslí, že tabulka č. 2 je ideálním řešením, kde se plýtváním sloupci dostáváme do vysoké normální formy tabulky, není zatím ještě dostatečně připraven na dekompozici v databázovém prostředí. Bohužel jsem podobných tabulek viděl už celou řadu, ať už jsem byl jejich tvůrcem nebo smutným pozorovatelem. Měli bychom navrhovat tabulky tak, aby počet sloupců nebyl přehnaný.

Do tabulky č. 3, která je náznakem skutečného řešení problému, můžeme klást rovněž obdobné dotazy. Stálo by za to si uvědomit, jak do této tabulky v 1NF formulovat dotaz, kteří trpaslíci jsou hodní. Dotaz by zněl:

```
SELECT TRPASLIK FROM TRAX WHERE
VLASTNOST='HODNY';
```

Takovýto dotaz vyvolá zobrazení křestních jmen hodných trpaslíků – Šmudly a Kejchala. Kdybychom se chtěli zeptat, kdo z trpaslíků uvedených v tabulce č. 3 se narodil roku 1830, potom nebudeme nadšeni. Příslušný dotaz

```
SELECT TRPASLIK FROM TRAX WHERE
ROKNAR=1830;
```

by totiž vygeneroval ne jeden, ale tři řádky a ve všech by bylo napsáno stejné křestní jméno Šmudla.

Obdobný trapas by nastal, pokud bychom se tázali, kterého roku se narodil Šmudla. Dotaz by zněl:

```
SELECT ROKNAR FROM TRAX WHERE
TRPASLIK='SMUDLA';
```

Opět by došlo k vygenerování tří řádků, ve kterých by byly stejné letopočty 1830. Zádrhel je totiž ve velké závislosti mezi sloupcem TRPASLIK a sloupcem ROKNAR, která nám kazí radost z tabulky č. 3. Není proto divu, že tabulka č. 3 je zatím jen v 1NF a bude třeba ji vylepšit. Ale abych vyslal také nějaký optimistický signál o správnosti návrhu, uvádím dotaz na všechny Šmudlovy vlastnosti:

```
SELECT VLASTNOST FROM TRAX WHERE
TRPASLIK='SMUDLA';
```

Dojde k vygenerování tří řádků, ve kterých jsou vyjmenovány všechny Šmudlovy vlastnosti, a to nikoli vedle sebe, jak to poskytuje tabulka č. 1, ale pod sebou. Takovou možnost nám tabulka č. 2 neposkytovala.

Pokud byste chtěli v předstihu studovat uvedené SQL dotazy, doporučuji vám opatřit si jakoukoli monografii o SQL jazyce nebo se rovnou zanořit do nápovědy příslušného SQL serveru.

Nezapomeň na klíč

V databázových systémech se při popisu tabulek nevyhneme zavedení pojmu jednoduchého a složeného klíče. Klíč nám slouží k rychlému určení toho správného řádku příslušné tabulky. Nejprve se budeme zabývat problematikou jednoduchého klíče, ale jeho definici si ponecháme až do dalšího dílu našeho seriálu.

Jaromír Kukal