edice PRŮVODCE


GRADA

# SQL

## Podrobný průvodce uživatele

Marek Laurenčík

- Úvod do relačních databází
- Tvorba dotazů příkazem SELECT
- Výpočty a funkce v dotazech
- Spojování více tabulek v dotazu
- Souhrny a seskupování
- Tvorba tabulek a práce s daty v tabulkách
- Příklady k procvičování

soubory ke stažení na WWW.GRADA.CZ


GRADA

# SQL

## Podrobný průvodce uživatele

Marek Laurenčík

Grada Publishing

**Marek Laurenčík**

# SQL
## Podrobný průvodce uživatele

Vydala Grada Publishing, a.s.
U Průhonu 22, Praha 7
obchod@grada.cz, www.grada.cz
tel.: +420 234 264 401, fax: +420 234 264 400
jako svou 6996. publikaci

Spoluautor Michal Bureš
Odpovědná redaktorka Věra Slavíková
Sazba Jan Šístek
Počet stran 216
První vydání, Praha 2018
Vytiskly Tiskárny Havlíčkův Brod, a. s.



ISBN 978-80-271-2155-7 (ePub)
ISBN 978-80-271-2154-0 (pdf)
ISBN 978-80-271-0774-2 (print)

# Obsah

# Úvod

Obsahem této knihy jsou základy tvorby dotazů v jazyku SQL (Structured Query Language). Jazyk SQL je základním prostředkem pro komunikaci s databázovými systémy. Kniha vychází ze standardu jazyka SQL s přihlédnutím k implementaci tohoto jazyka v několika nejčastěji používaných databázových systémech.

Kniha je určena především čtenářům, kteří chtějí s databázemi pracovat a potřebují se naučit základy jazyka SQL. Kromě čtenářů, kteří zpracovávají uložená data, může být užitečná i tvůrcům webových stránek a programátorům např. v jazyku Visual Basic for Application, neboť i v těchto činnostech je základní znalost jazyka SQL nezbytná.

Základní tématem knihy je příkaz SELECT, tedy vyhledávání a zpracování dat z databází. Toto téma je v knize probíráno z různých zorných úhlů včetně tipů, triků a případných problémů, se kterými se můžete při práci s databází setkat. Dalším probíraným tématem jsou akční dotazy, tedy zápis do tabulek, přidávání a mazání řádků v tabulkách. Tato dvě základní témata knihy jsou doplněna základy práce se strukturou tabulek. Administrace databázových systémů v knize zařazena není. Důvodem je omezený rozsah knihy a také skutečnost, že kniha není určena pro administrátory, ale především pro běžné uživatele, u kterých se dá předpokládat, že tuto problematiku řešit nebudou.

Kniha vychází ze standardu jazyka SQL. Součástí knihy je však řada příkladů k procvičení, a proto není možné odhlédnout od jednotlivých implementací tohoto jazyka v různých databázových systémech. Kniha se orientuje především na dva databázové systémy, široce využívané v České republice: Microsoft SQL Server a databázový systém MySQL. První z nich je základem zpracování dat zejména ve firmách a institucích menší a střední velikosti, druhý je využíván pro tvorbu webových stránek. Dalším důvodem pro tuto volbu bylo, že implementace jazyka SQL na MS SQL Serveru se příliš neodchyluje od standardu tohoto jazyka a že oba databázové systémy jsou zdarma k dispozici ke stažení na internetu.

Výklad jazyka SQL popisuje souběžně odchylky obou dvou uvedených databázových systémů od standardu a je doplněn odchylkami pro MS Access, databázový systém Oracle a databázovou knihovnu SQLite. Odchylky v implementacích jazyka SQL u jednotlivých databázových systémů jsou graficky zvýrazněny. Výjimkou je kapitola 7 (věnovaná standardním funkcím), které jsou u každého databázového systému jiné.

Jednotlivé kapitoly jsou vždy doplněny sérií příkladů pro samostatné řešení. Soubory, potřebné pro vypracování příkladů v databázích SQL Serveru a MySQL, naleznete na webových stránkách nakladatelství Grada (www.grada.cz), v sekci této knihy. V kapitole 13 jsou uvedena řešení jednotlivých příkladů. Tato řešení jsou spíše pro orientaci a kontrolu; jednotlivé problémy je možno velmi často řešit několika různými způsoby. Popis zkušební databáze, použité v příkladech a návod k její instalaci jsme zařadili do dodatku.

Protože tvorba dotazů v jazyku SQL je především tvůrčí prací, chtěli jsme knihu koncipovat tak, aby nebyla sbírkou hotových receptů, ale vedla čtenáře k samostatnému používání jazyka SQL v praxi.

V knize jsou použita následující typografická označení:

- Příkazy v kódu SQL jsou zvýrazněny `neproporcionálním písmem`.
- Konstanty jazyka SQL jsou zvýrazněny ***kurzívou***.
- Názvy souborů jsou zvýrazněny ***tučnou kurzívou***.
- Důležité pojmy a příkazy v uživatelských rozhraních jsou zvýrazněny **tučným písmem**.

Ať se Vám daří!
Autoři

# 1 Relační databáze

Co se v této kapitole dozvíte:

- Základní informace o databázích
- Strukturu databázového souboru

## 1.1 Co je databáze?

Tato kniha pojednává o práci s databázemi. Proto bychom měli začít definicí, co databáze vlastně je.

Nejobecnější definice databáze by mohla být ta, že se jedná o uspořádanou soustavu dat. Podle této definice nezáleží na tom, jestli jsou data uchovávána v papírové nebo elektronické podobě. Při bližším pohledu je zřejmé, že požadavek uspořádanosti dat vyžaduje, aby bylo možno soustavu dat snadno upravovat, doplňovat novými informacemi, popř. odstraňovat informace již zbytečné. Dalším požadavkem samozřejmě je, aby se potřebné informace snadno vyhledávaly. Pokud jsou informace uchovávány v papírové podobě, požadavek uspořádanosti bude těžko splňovat seznam údajů, napsaný na jedné nebo více stránkách za sebou. Naproti tomu data, uložená v papírové kartotéce, již požadavek uspořádanosti splňují.

Papírové kartotéky, které starší generace ještě pamatují, jsou předchůdci dnešních databází. Splňovaly hlavní databázové požadavky a práce s nimi připomínala v mnoha směrech práci s dnešními databázemi. Uchovávaná data bylo možné uspořádávat různými způsoby, do kartotéky bylo možné poměrně snadno přidávat nové položky. Veškeré operace se ovšem musely provádět manuálně a rozsáhlé kartotéky zabraly mnoho místa.

Proto se již před poměrně dlouhou dobou objevila snaha o strojní zpracování dat. Prvním prakticky použitelným systémem byly děrné štítky, které se zpracovávaly na elektromechanických strojích. Tímto způsobem bylo provedeno sčítání lidu ve Spojených státech v roce 1890.

Od počátku padesátých let minulého století se začaly vyvíjet samočinné počítače a bylo přirozené, že se záhy objevily snahy využít počítače pro hromadná zpracování dat. Programy pro první samočinné počítače byly vesměs psány ve strojovém kódu, který byl sice použitelný pro vědecké a technické výpočty, ale nevhodný pro zpracování dat. V roce 1959 se konala konference zástupců průmyslových a obchodních firem a amerického ministerstva obrany, která došla k závěru, že je nutno vytvořit univerzální databázový jazyk, vhodný pro zpracování dat. Výsledkem byl vývoj jazyka COBOL (Common Business-Oriented Language), který využívaly sálové počítače řady firem.

### Typy databázových modelů

Prvním databázovým modelem, vzniklým v šedesátých letech minulého století, byl hierarchický model. V tomto modelu jsou data uspořádána v hierarchické struktuře ve vztazích rodič – potomek.

Hierarchická koncepce databázového modelu však nemohla vystihnout všechny možnosti ve vztazích mezi uchovávanými daty. Proto byla záhy nahrazena síťovým modelem. Tento model v podstatě vychází z hierarchické koncepce, ale doplňuje jej o vztahy „více ku více", tedy vztah, kdy jeden potomek může mít více rodičů. Síťové databáze byly využívány poměrně dlouhou

dobu hlavně na sálových počítačích. Nejznámějším produktem tohoto typu byl databázový systém IDMS od firmy Cullinane Corp.

Další pokrok nastal roku 1970, kdy E. F. Codd uveřejnil článek „A Relational Data Model for Large Shared Data Banks", popisující koncepci relační databáze. Koncepce relační databáze je založena na uchovávání dat v tabulkách, ve kterých jsou data uspořádána do řádků a sloupců. Údaje, ukládané v některých sloupcích, jsou pro různé tabulky společné a slouží tak k vyjádření vzájemných vztahů (relací) mezi jednotlivými datovými tabulkami. Tato koncepce se ukázala jako velice plodná a tvoří základ i databázových systémů, používaných v dnešní době.

Pro práci s relačními databázemi vyvinula firma IBM první verzi dotazovacího jazyka SQL (Structured Query Language). Prvním databázovým systémem, využívajícím jazyk SQL, byl Oracle, využívaný na počítačích VAX. Záhy nato vyvinula firma IBM pro svoje počítače databázový software DB2.

Rozvoj osobních počítačů od osmdesátých let minulého století je úzce spojen s rozvojem relačních databází. Řada databázových systémů[1], využívajících koncepci relačních databází a jazyk SQL bylo vyvinuto už pro operační systém MS DOS (dBase, FoxPro, Paradox atd.). Rozvoj relačních databází pokračoval v souvislosti s nástupem počítačových sítí a internetu.

V České republice se v současné době můžete setkat s různými databázovými systémy, využívanými na osobních počítačích. K nejčastěji používaným systémům patří:

- Microsoft Access – databázový systém, určený pro menší evidence. Je součástí kompletu MS Office.
- Microsoft SQL Server – výkonný databázový systém, využívaný zejména ve středních a větších institucích.
- MySQL – databázový systém, který je k dispozici zdarma ke stažení nebo pro komerční účely prodávaný firmou Oracle Corporation. Je využíván zejména pro tvorbu webových stránek.
- Oracle – velice výkonný databázový systém, využívaný pro velké objemy dat.
- SQLite – tento systém není založen na principu klient – server, ale představuje knihovnu, využitelnou v řadě programovacích jazyků. Je využíván především na internetu, v mobilních telefonech apod. Jeho výhodou je jednoduchost, nevýhodou nízká výkonnost.

Některé možnosti relačních databází mají i poslední verze tabulkového kalkulátoru MS Excel (verze 2013 a 2016).

## 1.2 Relační databáze na osobním počítači

### Vlastnosti tabulek

Jak jste již poznali, v relační databázi jsou data uložena ve formě tabulek. První otázkou tedy je, jaká forma tabulky bude v relační databázi použitelná, neboli jaké požadavky musí tabulka splňovat, aby vykazovala „databázové vlastnosti".

Nejprve si řekněme, které tabulky, vytvořené např. v tabulkovém kalkulátoru, databázové vlastnosti nemají.

Příklad první (převzatý ze skutečné praxe): údaj ve sloupci „Splatnost" v seznamu faktur, který má tvar „Splatnost 5. 8. 2017, do 30. 9. 2017 nezaplaceno, upomínáno 12. 10. a 30. 10., 9. 11. 2017

1 Je vhodné rozlišovat pojem „databázový systém", což je software pro práci s daty a pojem „databáze", což je soubor dat, zpracovávaný databázovým systémem.

zaplaceno 14 000 Kč, zbytek do konce roku". Pokud by z tohoto textu měl nějaký databázový program získat např. datum splatnosti, jednalo by se o značně obtížný úkol.

Příklad druhý: tabulka rozdělená na několik částí tak, že se do ní vložily prázdné řádky. Ty sice mohou přispět k přehlednosti tabulky, ale pro počítačová zpracování jsou velkým problémem. Jedná se o součást tabulky, kde údaje nejsou známy, nebo se mají při zpracování vyřadit?

Příklad třetí: ve sloupci s peněžními částkami se vyskytují údaje typu „xxx" (což má znamenat, že údaj není k dispozici), ve sloupci s datumovými hodnotami[2] se vyskytují texty „červen 2017", „průběžně" atd. Při zpracování takovýchto sloupců by musel databázový systém neustále testovat, o jaký údaj se vlastně jedná.

Příklad čtvrtý: hlavička tabulky vypadá takto:

| **Leden** | | | | **Únor** | | | |
|---|---|---|---|---|---|---|---|
| ***Příjem*** | | ***Výdaj*** | | ***Příjem*** | | ***Výdaj*** | |
| Převodem | Hotově | Převodem | Hotově | Převodem | Hotově | Převodem | Hotově |

Po grafické stránce je hlavička pěkná, pro databázový systém nepoužitelná – jak se mají identifikovat jednotlivé sloupce?

Aby měla tabulka na listu databázový charakter, musí splňovat některé podmínky:

- Tabulka dat je souvislá, tj. neobsahuje prázdné řádky nebo sloupce. Některé buňky v řádku mohou být nevyplněné, avšak každý řádek v tabulce musí obsahovat aspoň jednu vyplněnou buňku, aby tabulkový kalkulátor mohl určit rozsah, kde jsou data umístěna.
- Každý sloupec v tabulce má samostatný nadpis, nadpisy u jednotlivých sloupců se musí vzájemně lišit. Nadpisy jsou umístěny ve společném řádku, řádek s nadpisy je jediný a tvoří první řádek tabulky.
- Údaje v jednotlivých sloupcích mají jednotný charakter: texty, čísla, datumové hodnoty nebo logické hodnoty PRAVDA a NEPRAVDA.

U databázových systémů jsou tyto požadavky automaticky zaručeny už tím, že má tabulka přesně definovanou strukturu a u každého sloupce je určen nejen jeho název, ale také charakter údajů, které lze do sloupce zapisovat. Jiný typ údaje se do sloupce zapsat nedá. Uvedené zásady je však rozumné dodržovat i při práci s tabulkovým kalkulátorem, zvláště tehdy, pokud chcete současně používat relační databázi. Při importování dat do relační databáze si tak ušetříte zbytečné problémy.

## Databázový software

Pro práci s relačními databázemi by bylo vhodné rozlišovat vlastní data a software, který jejich tvorbu a zpracování umožňuje. Pro obojí se často používá název „databáze"; hovoří se tedy o databázi prodejů a současně o databázi SQL Serveru. Proto pro databázový software budeme používat pojem „databázový systém", kdežto pojem „databáze" bude určen pro zpracovávaná data.

Databázový systém pracuje takto:

- Po instalaci softwarového balíku se na disku počítače nebo na síťovém serveru vytvoří tzv. databázový server. Jedná se v podstatě o instalované programové vybavení, nikoliv o počítačovou jednotku.

2 Výraz „datumový" není součástí spisovné češtiny, ale budeme jej v této knize používat pro lepší srozumitelnost, abychom odlišili hodnoty typu „datum" od pojmu „data jako taková".

- Po spuštění databázového serveru je nutné se k němu přihlásit. Je to možné provést v zásadě dvěma způsoby. Při prvním je oprávněnost přístupu zajištěna již tím, že se uživatel přihlásil do počítačové sítě. U druhého způsobu je nutné zadat přihlašovací jméno a heslo.
- Databázový server umožňuje pracovat s několika soubory dat – několika databázemi. Nejprve je nutné databázi založit, vytvořit v ní jednotlivé tabulky a naplnit je daty. Databáze je uložena na disku jako jediný soubor nebo jako systém souborů, vzájemně propojených.
- Vlastní zpracování dat provádí část databázového software, označovaná jako „databázový stroj" (database engine). Komunikaci s databázovým strojem zajišťuje programové rozhraní, které může být jazykově upraveno. Tato komunikace probíhá prostřednictvím programu (např. u MS SQL Serveru program MS SQL Server Management Studio) nebo pomocí webové stránky (u systému MySQL stránka PHP MyAdmin).

Výjimku tvoří program MS Access, u kterého se není třeba přihlašovat k databázovému serveru. Postačuje spustit program a otevřít zpracovávanou databázi.

## 1.3 Struktura databázového souboru

Databázový soubor obsahuje sérii objektů různého typu. Hlavní typy objektů v databázovém souboru jsou tyto:

- **Tabulky** (tables): obsahují vlastní zpracovávaná data. Každá tabulka v databázi tvoří samostatný objekt a má přesně definovanou strukturu. Strukturou databázových tabulek se podrobně zabývá následující kapitola.
- **Pohledy** (views): jedná se o uložené příkazy jazyka SQL, které je možné opakovaně spustit a není nutné je vytvářet pokaždé znovu. V programu MS Access se pohledy označují jako **dotazy** (queries).
- **Diagramy**: tabulky v databázi jsou mnohdy na sobě závislé a jsou mezi nimi vzájemné vztahy, zajišťované pomocí údajů, společných pro dvě nebo více tabulek. Tyto vztahy je možné trvale uložit v objektu, označovaném jako diagram nebo relace.

# 2 Struktura databáze a tabulek

Co se v této kapitole naučíte:

- Typy polí v tabulkách
- Typy indexů
- Vlastnosti polí v tabulkách
- Relace mezi tabulkami
- Zásady pro správnou tvorbu databáze

Poznatky v této kapitole jsou pro vás užitečné i v případě, když nebudete datové tabulky sami vytvářet a budete je pouze využívat. Abyste mohli jazyk SQL efektivně používat, potřebujete mít aspoň základní představu o tom, jak databázový systém uložená data zpracovává.

Jak jsme si již uvedli, datové tabulky obsahují primární informace a tvoří proto základní kostru celé databáze. Databázová tabulka se svojí strukturou podobá tabulce v tabulkovém kalkulátoru: obsahuje jeden nebo více řádků; každý řádek obsahuje zpravidla několik údajů různého typu, které jsou rozděleny do sloupců. Sloupec v jednom řádku představuje v databázi základní jednotku informace.

V české literatuře se u databázových tabulek také používají pojmy „záznam" a „pole". V této knize budeme používat pojmy **řádek** (row) a **sloupec** (column), které jsou v souladu s terminologií, používanou standardem jazyka SQL.

## 2.1 Typy sloupců

Nejčastějšími typy údajů, které se ukládají v databázové tabulce, jsou textové, číselné, datumové a logické hodnoty (Ano/Ne nebo ***True***/***False***). Do tabulek v relační databázi je možné ukládat i komplikovanější údaje, jako jsou obrázky a další soubory, geografické údaje atd.

Jedním ze základních rysů databázové tabulky je, že se v jednom sloupci mohou ukládat pouze údaje stejného typu. Typ ukládaného údaje je určen při zakládání nové tabulky. Např. do sloupce, které je ve struktuře tabulky deklarováno jako datumové, nemůžete zapsat číslo nebo text. Z hlediska praktické práce je toto omezení nepatrné, ale důsledkem je mnohonásobně vyšší výkonnost při manipulaci s uloženými údaji: na rozdíl od tabulkového kalkulátoru nemusí databázový systém neustále zjišťovat, jaký typ údaje je právě zpracováván.

Jednoduché dělení údajů na textové, číselné, datumové a logické však nestačí. V relačních databázích se rozlišují celočíselné hodnoty a čísla s desetinnými místy. Rovněž textové a datumové údaje mohou být rozdílného typu. Při použití jazyka SQL k výběru informací z tabulky se např. textová pole různého typu mohou chovat rozdílně. Databázové systémy zpravidla používají tyto základní typy sloupců:

- Celočíselné typy.
- Typy s desetinnými čísly.
- Binární typy (0 nebo 1), fungující jako Ano (True) nebo Ne (False).
- Datumové hodnoty.
- Časové hodnoty.
- Datum a čas.
- Textové údaje s pevnou délkou.
- Textové údaje s proměnnou délkou.

## Číselné typy

U MS SQL Serveru se používají tyto číselné typy:

- TINYINT – kladné celé číslo v rozsahu 0 až 255.
- SMALLINT – celé číslo v rozsahu ± 32 767.
- INT – celé číslo v rozsahu ± 2 147 483 647.
- BIGINT – celé číslo v rozsahu ± 9 223 372 036 854 775 808.
- REAL – bez omezení desetinných míst s rozsahem cca $\pm 3{,}4.10^{38}$.
- FLOAT – bez omezení desetinných míst s rozsahem cca $1{,}8.10^{308}$.
- MONEY – desetinné číslo, kde je automaticky nastaveno zaokrouhlení na čtyři desetinná místa.
- DECIMAL(m, n), kde m je celková délka údaje (počet znaků) a n počet desetinných míst. Číselný údaj má rozsah $\pm 3{,}4.10^{38}$. Zadávaný údaj se automaticky zaokrouhlí.

Systém MySQL má tyto číselné datové typy:

- TINYINT – celé číslo v rozsahu ± 128.
- SMALLINT – celé číslo v rozsahu ± 32 767.
- MEDIUMINT – celé číslo v rozsahu ± 8 388 608.
- INT – celé číslo v rozsahu ± 2 147 483 647.
- BIGINT – celé číslo v rozsahu ± 9 223 372 036 854 775 808.
- FLOAT – bez omezení desetinných míst s rozsahem cca $\pm 3{,}4.10^{38}$.
- DOUBLE – bez omezení desetinných míst s rozsahem cca $1{,}8.10^{308}$.
- DECIMAL(m, n), kde m je celková délka údaje (počet znaků) a n počet desetinných míst. Číselný údaj má rozsah $\pm 3{,}4.10^{308}$. Zadávaný údaj se automaticky zaokrouhlí.

Číselné datové typy u MS Accessu:

- Bajt – kladné celé číslo v rozsahu 0 až 255.
- Celé číslo – celé číslo v rozsahu ± 32 767.
- Dlouhé celé číslo – celé číslo v rozsahu ± 2 147 483 647.
- Jednoduchá přesnost – bez omezení desetinných míst s rozsahem cca ± $3{,}4.10^{38}$.
- Dvojitá přesnost – bez omezení desetinných míst s rozsahem cca $1{,}8.10^{308}$.
- Desetinné číslo – údaj má rozsah ± $10^{38}$. Nadbytečná desetinná místa se odříznou.
- Měna – desetinné číslo zobrazené v měnovém formátu s volitelným počtem desetinných míst.

Novější verze Oraclu používají jediný datový typ NUMBER, který nahrazuje ostatní číselné typy. Tento datový typ je možné použít trojím způsobem:

- NUMBER bez parametrů je číslo s plovoucí desetinnou čárkou a přesností na 38 míst.
- NUMBER(p) je celočíselná hodnota s délkou (počtem číslic) p.
- NUMBER(p,s) je desetinní číslo s délkou p a počtem desetinných míst s.

Dále je možné používat ještě číselné typy INTEGER, SMALLINT, REAL, FLOAT, DECIMAL, které jsou v databázovém systému Oracle zařazeny pro zpětnou kompatibilitu.

## Textové typy

Textové typy u SQL Serveru:

- Typ CHAR – u tohoto datového typu platí, že pokud je textový řetězec kratší, než zadané maximum, zbytek textu je automaticky doplněn mezerami.
- Typ VARCHAR – tento textový typ si zaznamená počet zapsaných znaků, zapsaný text je bez koncových mezer.
- VARCHAR(MAX) – slouží pro uložení rozsáhlých textových údajů.

Protože u datového typu CHAR mohou být na konci mezery kdežto u typu VARCHAR nikoliv, při použití jazyka SQL se oba datové typy liší svým chováním.

Kromě uvedených datových typů jsou k dispozici ještě textové datové typy, začínající písmenem „N", tedy NCHAR, NVARCHAR, nvarchar(MAX). Tyto datové typy jsou určeny pro zápis textu v kódu Unicode, tedy pro případ textu psaného jinak než latinkou.

Textové typy u MySQL:

- Typ CHAR – textový typ s pevně zadanou délkou, ale zbytek textu není automaticky doplněn mezerami.
- Typ VARCHAR – stejný datový typ jako u SQL Serveru.
- TEXT – slouží pro uložení rozsáhlých textových údajů.

Také v MySQL jsou k dispozici analogické textové typy, určené pro zápis textu v kódu Unicode a začínající písmenem „N".

Textové typy u MS Accessu:

- Krátký text – textový údaj pevné délky, maximální délka 255 znaků. Zbytek textu není doplněn mezerami.
- Dlouhý text (dříve označovaný jako typ Memo) – textový údaj proměnné délky.

Textové typy u Oracle:

- CHAR (popř. NCHAR) – obdobný datový typ jako u SQL Serveru.
- VARCHAR2 (popř. NVARCHAR2) – obdobný datový typ jako u SQL Serveru.
- TEXT – slouží pro uložení rozsáhlých textových údajů.
- VARCHAR (popř. NVARCHAR) – tyto typy se považují v novějších verzích za zastaralé a jsou zařazeny pro zpětnou kompatibilitu.

### Datum a čas

Datumové a časové typy v SQL Serveru a MySQL:

- DATETIME – typ pro zápis datumu a času. Pokud do pole tohoto typu vložíte pouze datum, čas se zapíše jako „00:00:00".
- DATE – typ pro zápis datumové hodnoty. Při vložení datumu společně s časem se časový údaj zahodí.
- TIME – typ pro zápis času. Při vložení datumu společně s časem se zahodí datum.
- TIMESTAMP – časové razítko. Při vhodném nastavení výchozí hodnoty (viz dále) se u nového záznamu vloží do pole tohoto typu aktuální datum a čas.

Program MS Access má pro zápis datumu a času k dispozici datový typ „Datum a čas".

Datumové a časové typy v Oracle:

- DATE – typ pro zápis datumu a času.
- TIMESTAMP – časové razítko.

### Logické hodnoty

V SQL Serveru a MySQL je pro zápis logických hodnot určen datový typ BIT s hodnotami 1 (***True***) nebo 0 (***False***).

V programu MS Access je datový typ „Ano/Ne", ve kterém je hodnota ***True*** uložena jako –1 a hodnota ***False*** jako 0.

V databázi Oracle je pro zápis logických hodnot určen datový typ NUMBER(1).

### Datové typy v databázi SQLite

> Databáze SQLite se svým pojetím datových typů výrazně odlišuje od ostatních databázových systémů. V databázi se používá pouze několik málo datových typů, a databázový stroj provádí automatické konverze zadávaných dat. V zásadě mohou být uložena jakákoliv data do sloupce jakéhokoliv datového typu. Výjimkou je celočíselný primární klíč (viz dále). Běžně používanými datovými typy jsou INTEGER, TEXT a REAL. Datumové a časové hodnoty se mohou ukládat jako text nebo jako hodnota typu REAL.

## 2.2 Indexy v tabulkách

Jednotlivé řádky v tabulce jsou uloženy v pořadí, jak byly vytvářeny. Pokud potřebujete řádky v tabulce zobrazit v určitém pořadí, např. abecedně podle některého textového sloupce, databázový stroj musí napřed vybrat texty začínající písmenem „a", potom texty začínající písmenem „b" atd. U rozsáhlých tabulek může být toto seřazení časově náročné. Bylo by jistě daleko efektivnější, kdyby bylo v tabulce zaznamenáno také pořadí, v jakém mají být jednotlivé hodnoty v poli seřazeny. Tato informace o pořadí hodnot se označuje jako **index** nebo **klíč**. Je možné si jej představit jako další skrytý sloupec, ve kterém je pořadí zaznamenáno:

| pole | index |
|---|---|
| Praha | 3 |
| Aš | 1 |
| Zlín | 4 |
| Dolní Bousov | 2 |

Použití indexu u některého sloupce poněkud zpomaluje úpravu dat. Na druhé straně výrazně zrychluje výběry z tabulky pomocí jazyka SQL.

V relačních databázích se používají dva základní typy indexů:

- **Regulární** – hodnoty v indexovaném sloupci se mohou opakovat.
- **Unikátní** – hodnoty v indexovaném sloupci se nemohou opakovat.

Příkladem může být tabulka firem: ve sloupci se sídlem firmy se jistě mohou hodnoty opakovat, a proto použijete regulární index. Naproti tomu IČ je hodnota u každé firmy jedinečná a proto je vhodné použít pro tento sloupec index unikátní. Použití unikátního indexu zároveň zabrání, aby se v tomto sloupci vytvořily duplicitní hodnoty.

### Primární klíč

Každá tabulka by měla mít sloupec, který jednoznačným způsobem charakterizuje záznam. U tohoto sloupce použijete další typ indexu, označovaného jako **primární klíč**. Jedná se v podstatě o unikátní index, který však může být v tabulce pouze jeden (unikátních indexů může být i více).

Tabulka bez primárního klíče může sice existovat, ale je velmi rozumné takovouto tabulku vůbec nevytvářet. Jsou pro to následující důvody:

- Zpracování tabulky s primárním klíčem je efektivnější.

- Primární klíče jsou nezbytné pro tvorbu vzájemných vztahů mezi tabulkami.
- Ve většině databázových systémů včetně MS SQL Serveru a systému MySQL je existence primárního klíče nutná pro práci s hodnotami v tabulce. Do tabulky, vytvořené bez primárního klíče můžete sice přidávat nové řádky, ale možnost úpravy nebo mazání vytvořených údajů je omezená nebo znemožněná.

Primární klíč je možné vytvořit i ze dvou nebo více sloupců. V tabulce faktur je v jednom sloupci zaznamenán rok a ve druhém číslo faktury v rámci určitého roku:

| cislo | rok |
|---|---|
| 1 | 2017 |
| 2 | 2017 |
| 1 | 2018 |
| 2 | 2018 |

Bylo by možné přidat do tabulky další sloupec, obsahující například čísla 1, 2, 3, 4 atd. a tento sloupec použít jako primární klíč. Vytvořit složený primární klíč ze dvou sloupců (číslo faktury a rok) je však výhodnější: tabulka nemusí obsahovat další sloupec a existence primárního klíče vytvořeného ze dvou sloupců, zajistí, že se ve dvou různých řádcích nemůže opakovat číslo faktury a rok.

## 2.3 Hodnota NULL

Při práci s Excelem nebo jiným tabulkovým kalkulátorem jste zvyklí na to, že se prázdná buňka chápe ve vzorcích jako nula. V relačních databázích toto neplatí.

Ve sloupcích s číselnými nebo datumovými hodnotami není možné nic nezadat a při zapisování do tabulky nechat políčko prázdné. Místo toho mají relační databáze k dispozici konstantu ***NULL***, což znamená „nevím", „neznámo". Tuto konstantu však musíte do pole zadat.

Hodnota ***NULL*** znamená něco úplně jiného než prázdná buňka v Excelu:

- Jestliže je hodnota ***NULL*** zapsána v některém z číselných sloupců a tento sloupec použijeme při tvorbě vzorce, výsledkem výpočtu je vždy hodnota ***NULL***. Totéž platí i pro vzorce, využívající datumové nebo textové hodnoty.
- Protože hodnota ***NULL*** znamená „nevím", dvě hodnoty ***NULL*** se mezi sebou nerovnají. Tedy výraz „5 = 5" představuje hodnotu ***True*** (výraz platí), výraz „5 = 7" představuje hodnotu ***False*** (výraz neplatí), ale výraz „***NULL*** = ***NULL***" nepředstavuje ani hodnotu ***True***, ani hodnotu ***False***, nýbrž opět hodnotu ***NULL*** (pokud hodnoty neznáme, nemůžeme je mezi sebou porovnávat).

Výskyt hodnot ***NULL*** v některém sloupci tedy může práci s tabulkou dost nepříjemně zkomplikovat. Jak se s tímto problémem vyrovnat se dozvíte podrobněji v kapitolách o filtraci záznamů a o tvorbě vzorců.

### Hodnota NULL v textovém sloupci

Textové sloupce mohou také obsahovat hodnoty ***NULL***. Na rozdíl od číselných nebo datumových sloupců je možné údaj v tomto sloupci vymazat, takže připomíná prázdnou buňku. V tabulce však není uložena hodnota ***NULL***, nýbrž prázdný textový řetězec, tj. řetězec nulové délky, tedy text „". Relační databáze oba typy údajů, hodnotu ***NULL*** a prázdný textový řetězec důsledně