# Relace

V relační databázi jsou jednotlivé tabulky propojeny pomocí **relací**. Hlavním účelem relací mezi tabulkami je omezení výskytu redundantních (nadbytečných) dat. Data by se neměla v různých tabulkách v rámci jedné databáze opakovat. Relace je postavena na vazbě stejných hodnot mezi unikátním polem (primárním klíčem) jedné tabulky a odpovídajícím polem jiné tabulky. Pro vytvoření relace mezi dvěma tabulkami je nutné mít v každé tabulce speciální pole. V jedné z těchto tabulek je to **primární klíč**. (v případě relace 1:1 může být primární klíč v obou tabulkách). Takovou tabulku označujeme jako **primární tabulku**. Ve druhé tabulce vytváříme speciální pole pro účely relace. Toto pole označujeme jako cizí klíč, obsahuje hodnoty primárního klíče z jiné tabulky. Tabulka obsahující **cizí klíč** je označována jako **sekundární tabulka**.

Pole primárního klíče obsahuje pouze unikátní hodnoty. Pole cizího klíče může obsahovat i stejné hodnoty. **Pole primárního i cizího klíče musí obsahovat** pro správné vytvoření relace **stejné hodnoty**.

Rozlišujeme tři základní typy relací:

* **Relace typu 1:1** (výjimečný) – jednomu záznamu primární tabulky odpovídá maximálně jeden záznam v sekundární tabulce. *Příklad:* jedné osobě je přiděleno právě jedno rodné číslo.
* **Relace typu 1:N** (nejčastější) – jednomu záznamu primární tabulky odpovídá jeden nebo více záznamů v sekundární tabulce. *Příklad:* jeden zákazník si objednal několik našich výrobků.
* **Relace typu M:N** (velmi často) – jednomu nebo více záznamům primární tabulky odpovídá jeden nebo více záznamů v sekundární tabulce. Tyto relace řešíme pomocí spojovací tabulky, kterou pomocí dvou relací typu 1:N propojíme s původními tabulkami. *Příklad:* máme dvě tabulky Osoby a Cesty. Zaměstnanec mohl vyjet na více služebních cest. Na jednu služební cestu mohlo jet více zaměstnanců. V databázi je nutné vytvořit můstkovou tabulku, která převede jednu vazbu M:N na dvě vazby 1:M a 1:N. Vytvoříme tedy tabulku Výjezdy, která bude ve vazbě 1:M k tabulce Osoby a ve vazbě 1:N k tabulce Cesty.

**Referenční integrita**

Referenční integrita udržuje neporušenost relací mezi tabulkami. Nedovolí nám vložit do sekundární tabulky záznam, který by neměl odpovídající záznam v primární tabulce. Dále si pohlídá si změnu hodnot cizího klíče při změně primárního klíče. V rámci referenční integrity je možné nastavit pravidla pro odstraňování záznamů.

**Příklad (zatím bez vytváření v Accessu):**

**Jak zaznamenávat data do databáze?**

**Jedna informace jeden řádek:**  
Jan Vopršálek, 12. 12. 1969, Praha, generální ředitel,  
Pepa Zdepa, 11. 11. 1972, New York, bezdomovec

Máte-li data na řádku, zdají se nepřehledná. Proto Vás napadne uspořádaní do tabulek.

|  |  |  |  |
| --- | --- | --- | --- |
| **Jmeno** | **Narozen** | **Bydliste** | **Povolani** |
| Jan Vopršálek | 12. 12. 1969 | Praha | generální ředitel |
| Pepa Zdepa | 11. 11. 1972 | New York | bezdomovec |

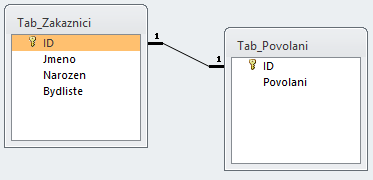
Pokud budeme chtít, je možné od sebe oddělit záznamy o osobách a rozložit je např. do dvou tabulek. Vznikne tak relace 1:1. Jednomu záznamu v tabulce Zákazníci bude odpovídat jeden záznam v tabulce Povolání. Dosáhneme tak toho, že povolání budou sledována na jiném místě, např. z důvodu zajištění ochrany údajů, pokud se jedná o citlivou informaci. K první tabulce může mít přístup více lidí, zatímco k druhé pouze oprávněné osoby. Relace 1:1 se také využívá v případě, kdy určité údaje má jenom malá skupina ze sledovaných záznamů a bylo by tak zbytečné pro ně zakládat většinou prázdné pole u každého záznamu v jedné tabulce.

**Tabulka Zákazníci**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Jmeno** | **Narozen** | **Bydliste** |
| 1 | Jan Vopršálek | 12. 12. 1969 | Praha |
| 2 | Pepa Zdepa | 11. 11. 1972 | New York |

**Tabulka Povolání**

|  |  |
| --- | --- |
| **ID** | **Povolání** |
| 1 | generální ředitel |
| 2 | bezdomovec |



**Příklad: 6. Relace 1-1.accdb**

Představte si, že dané osoby jsou zákazníci, kteří kupují naše výrobky a že je máme stále v jedné tabulce. Napadne Vás do tabulky přidat další sloupec a bude vypadat, že máte vyřešeno? Ukažme si příklad:

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Jmeno** | **Narozen** | **Bydliste** |
| **Výrobek 1** | Jan Vopršálek | 12. 12. 1969 | Praha |
| **Výrobek 1** | Pepa Zdepa | 11. 11. 1972 | New York |
| **Výrobek 2** | Pepa Zdepa | 11. 11. 1972 | New York |
| **Výrobek 3** | Pepa **Y**depa | 11. 11. 1972 | New York |

Takže máme přehlednou tabulku a vidíme, že ač Pepa Zdepa, koupil 3 výrobky, naše tabulka nám při filtrování ukáže jen dva, jelikož při vyplňování tabulky došlo k překlepu (Y/Z) a již máme pro tabulku tři různé osoby. Navíc se nám v tabulce objevují stále stejné záznamy. Což je nevýhodné (zabírá v tabulce místo a čas pro vyplňování). Proto využijeme relace.

Správně vytvořená databáze nám umožňuje vytvořit logické vazby mezi tabulkami. Budeme pokračovat v našem příkladu. Rozdělíme ho do dvou tabulek. Jedna se bude jmenovat **Zákaznici** a druhá **Výrobky**.

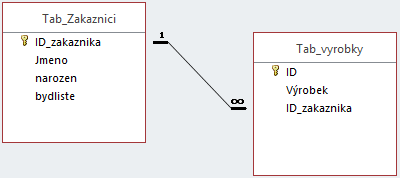
**Tabulka Zákaznici**

| **id zákazníka** | **Jmeno** | **Narozen** | **Bydliste** |
| --- | --- | --- | --- |
| 1 | Jan Vopršálek | 12. 12. 1969 | Praha |
| 2 | Pepa Zdepa | 11. 11. 1972 | New York |

**Tabulka Výrobky**

| **Výrobek** | **id zakazníka** |
| --- | --- |
| Výrobek 1 | 1 |
| Výrobek 1 | 2 |
| Výrobek 2 | 2 |
| Výrobek 3 | 2 |

Takže již jsou patrné výhody. Tabulky se zjednodušily. Jakmile si náš zákazník koupí další výrobek přidáme do tabulky Výrobky další záznam. Vytvořili jsme právě relaci **1:N.**

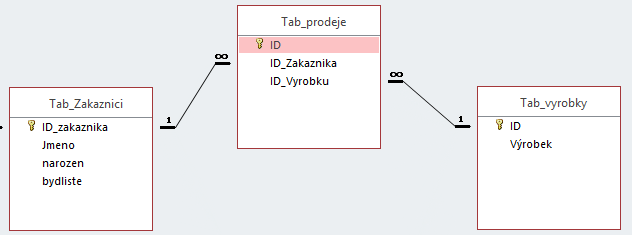


**Příklad: 6. Relace 1-N příklad.accdb**

Stále však dochází k redundanci dat, **protože v tabulce Výrobky se nám opakuje jméno Výrobku (v praxi bychom navíc sledovali nejenom jméno výrobku, ale jistě i spoustu dalších údajů o něm a tyto údaje by se nám stále opakovaly). Proto je vhodné v tabulce Výrobky mít každý výrobek uveden pouze jednou, a protože se bude (jako podnikatelé v to alespoň doufáme) prodávat opakovaně více zákazníkům, je nutné založit spojovací tabulku, kterou nazveme například Prodeje. Mezi zákazníkem a výrobky pak vznikne vztah M:N, díky dvěma vazbám 1:N na spojovací tabulku Prodeje**.

**Tabulka Prodeje**

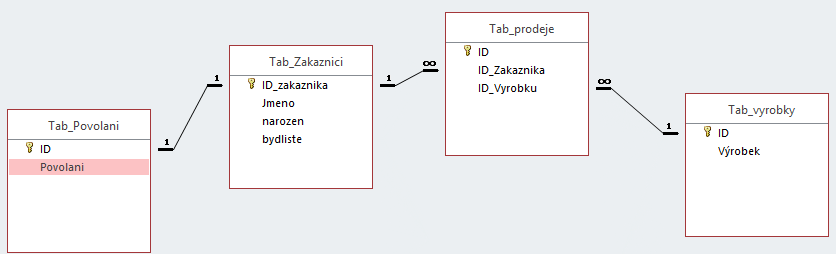
| **ID prodeje** | **ID zákazníka** | **ID Výrobku** |
| --- | --- | --- |
| 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 2 | 2 |
| 4 | 2 | 3 |



**Příklad: 6. Relace M-N příklad.accdb**

Platí tedy, že více kterýchkoliv zákazníků, si může koupit více kterýchkoliv výrobků.

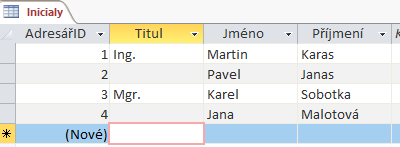
Závěrečný přehled celé situace i s relací 1:1, která vedla k oddělení povolání:



## Relace 1:1 – Příklad Lidé

Zadání:

1. Vytvořte databázi **Lide**, která bude obsahovat tabulky **Inicialy** a **Adresy.**
2. Tabulku **Inicialy** bude obsahovat sloupec **AdresářID** s automatickým číslováním a bude primárním klíčem tabulky. Dále bude obsahovat sloupce **Titul** (krátký text, velikost pole 10), **Jméno** (krátký text, velikost pole 20), **Příjmení** (krátký text, velikost pole 20). Tabulku vyplňte následujícími údaji.



1. Tabulku **Adresy** vytvořte importem z Excelu (soubor **Adresy.xlsx**). Primární klíč bude přidán automaticky.
2. Tabulky propojte relací 1:1 pomocí sloupce **ID**. Referenční integritu nezatrhávejte.
3. Přidejte do tabulky **Inicialy** nový záznam. Poté tabulku uložte.

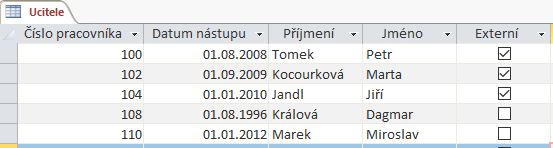


1. Vytvořte dotaz, který z tabulky **Inicialy** vybere sloupce **Titul** a **Příjmení** a z tabulky **Adresa** sloupec **Město**. Vyzkoušejte v tomto dotazu **Typ spojení** relace **1**, **2** a **3** a zobrazte si vždy výsledky v dotazu. Dotaz nazvěte **Vyber-D**.
2. Vytvořte formulář s podformulářem pro zápis nových pracovníků. Hlavní formulář nazvěte **Dotaznik** a vedlejší formulář **Informace**. Vyzkoušejte si pomocí tohoto formuláře prohlížení stávajících pracovníků i zápis nových, včetně jejich adres.

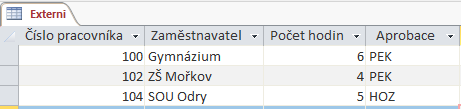
## Relace 1: 1 - Příklad Učitelé

Zadání:

1. Vytvořte databázi **Obchodka**, která bude obsahovat tabulky **Ucitele** a **Externi**.
2. Tabulku **Ucitele** vytvořte pomocí návrhového zobrazení, bude mít tato pole a údaje:



1. Jako primární klíč definujte **Číslo pracovníka**.
2. Tabulku Externi vytvořte také pomocí návrhového zobrazení s těmito poli a údaji. **Číslo pracovníka** bude zadáváno pomocí rozbalovací nabídky o dvou sloupcích (Číslo pracovníka a Příjmení), údaje budou vybírány z tabulky **Ucitele**. Zajistěte, aby se jednalo pouze o **externí pracovníky**. Primární klíč nastavte na sloupec **Číslo pracovníka**.



1. Zkontrolujte automatické vytvoření relace 1:1 mezi tabulkami.
2. Přidejte další pracovníky do tabulky **Ucitele**.

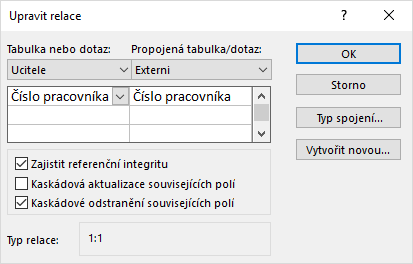


1. U učitelky Jany Laškové, která je externí pracovnicí, zadejte zaměstnavatele **Gymnázium**, počet hodin **6** a aprobaci **EKO**. Všimněte si, že tato osoba je již v rozbalovací nabídce Čísla pracovníků.
2. Vyzkoušejte, že je možné do tabulky **Externi** vložit záznam (učitele) s vymyšleným číslem, tj. takového, který není v tabulce **Ucitele**, např.



Záznam pak smažte a upravte relaci tím, že aktivujete zatržítko **Zajistit referenční integritu**. Po jeho aktivaci nebude možné přidat záznam do tabulky **Externi**, aniž by byl definován v tabulce **Ucitele**.

1. Vyzkoušejte, že nelze odstranit v tabulce **Ucitele** např. záznam **Tomek**, protože má související záznam v tabulce **Externi**. Pokud chcete odstranit např. záznam **Tomek** tak, aby se odstranil i odpovídající záznam v tabulce **Externi**, je třeba aktivovat v okně s nastavením relací položku **Kaskádové odstranění souvisejících polí**.



Stejně tak nelze měnit **Číslo pracovníka** v tabulce **Ucitele**, pokud není zatržena volba **Kaskádová aktualizace souvisejících polí**.

Zkuste u záznamu **104** změnit číslo na **105** – **nelze uložit**.

Pokud zatrhnete **Kaskádová aktualizace souvisejících polí**, bude možné **Číslo** změnit a zároveň se odpovídající číslo změní i v tabulce **Externi**.

1. Vytvořte dotaz, který z tabulky **Ucitele** vybere **Příjmení** a **Jméno** **pracovníka** a z tabulky **Externi** pole zaměstnavatel. Zobrazte **všechny** učitele. Dotaz nazvěte **Externiprac-D**.

## Relace 1:N – příklad Škola

1. Zkopírujte databázi **Skola** na disk H:
2. Vytvořte pomocí návrhového zobrazení novou tabulku **Tridy**. V ní budou uvedeny slupce **Zkratka** (**primární klíč**) a **Tridni**. Tabulku doplňte těmito údaji:



1. Vytvořte relaci mezi tabulkou **Tridy** a **Zaci** s tím, že zajistíte referenční integritu i aktualizaci polí.
2. Vytvořte formulář s podformulářem pro zadání nových žáků. Formulář bude splňovat tyto parametry:
   1. Formulář bude obsahovat pole **Zkratka** a **Třídní** z tabulky **Tridy**.
   2. Bude obsahovat pole **Jméno**, **Příjmení**, **Datnar**, **Hodiny** a **Prospěch** z tabulky **Zaci**.
   3. V záhlaví formuláře vytvořte nadpis **Obchodní akademie**.
   4. Na formuláři vytvořte příkazové tlačítko **Přidat třídu** a tlačítko **Zavřít formulář** s obrázkem dveří.
   5. Formulář uložte pod názvem **Prehled**.
3. Nyní do formuláře přidejte další třídu. Bude to **1. D**, třídní učitel **Novotná**.
4. Vytvořte dotaz (viz. obrázek níže), který zobrazí zkratku třídy, třídního učitele a počet žáků ve třídě. Zajistěte, že zde bude uvedena i 1. D, která je zatím bez žáků. Sloupec s počty žáků bude nazván **Počet žáků**. Dotaz nazvěte **Pocet-D**.

****

1. Zjistěte pomocí křížového dotazu, kolik žáků se narodilo v jednotlivých letech a v jednotlivých třídách. Dotaz nazvěte **Krizovy-D**.



1. Vytvořte dotaz, jenž po vyžádaném zadání **Příjmení** a **Jména** studenta připočte každému studentovi 5 hodin absence. Dotaz nazvěte **Zameskane-D**.
2. Importujte soubor **1D.xlsx** do nově vytvořené tabulky **Trida1D** bez primárního klíče.
3. Vytvořte přidávací dotaz, který přidá záznamy z tabulky **Trida1D** do tabulky **Zaci**. Dotaz nazvěte **Pridej-D**. Poté dotaz spusťte.
4. Vytvořte sestavu, která zobrazí:
   1. **Jména** a **Příjmení** žáků, počet zameškaných hodin, průměrný prospěch.
   2. Za celou třídu součet hodin a průměrný prospěch, zaokrouhlený na 2 desetinná místa.
   3. Celkový počet žáků v prvních ročnících – v zápatí sestavy.
   4. Sestavu pojmenujte **Prehled**.

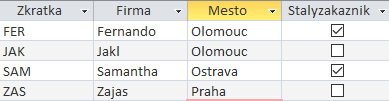
## Relace 1:N – příklad Grily

(Navazuje pak i na relaci N:N)

Zadání:

Zkopírujte databázi Grily na disk H:

1. Vytvořte dotaz, který vytvoří novou tabulku s názvem **Odberatele**. V ní bude jediný sloupeček **Odberatel** z tabulky **Prodej** a bude obsahovat jedinečné hodnoty. Název dotazu bude **Odb1-D**. Dotaz po vytvoření spusťte.
2. Upravte tabulku **Odberatele** tak, že doplníte sloupečky **Firma** (typ pole bude krátký text, velikost pole bude 20 znaků), **Město** (typ pole krátký text, velikost pole 15 znaků), **Stalyzakaznik** (typ pole ANO/NE). Dále přejmenujte položku **Odberatel** na **Zkratka**.
3. Vložte do tabulky tyto údaje:

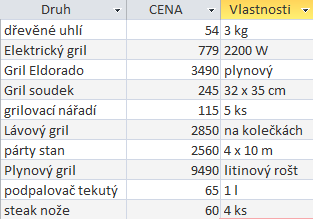


1. U pole **Zkratka** nadefinujte primární klíč.
2. Pro přidání či změnu dalších záznamů zajistěte vytvoření rozbalovací nabídky u pole **Odberatel** v tabulce **Prodej** (jako zdroj dat nabídky vyberte tabulku **Odberatele**, zobrazovat se budou sloupečky **Zkratka a Firma**).
3. Upravte relaci mezi poli **Zkratka** z tabulky **Odberatele** a polem **Odberatel** z tabulky **Prodej** tak, aby do tabulky **Prodej** nemohl být přidán žádný záznam s odběratelem, který neexistuje v tabulce **Odberatele**. Zajistěte možnost případné změny zkratky **Odberatele**.
4. V tabulce **Odberatele** změňte **Zkratku** firmy (odběratele) **FER** na **FEO**.
5. Vytvořte dotaz, který zjistí součet prodaného množství v jednotlivých městech. Dotaz nazvěte **Suma-D**.
6. Vytvořte dotaz, který po vyžádaném zadání data **od do** zobrazí **Název** prodaného zboží a **Firmu**. Záznamy budou seřazené podle sloupce **Datum** vzestupně. Dotaz nazvěte Kalendar-D. Zkuste zadat termín 1. 12. – 31. 12. 2018.
7. Vytvořte sestavu, která umožní přehled nákupů u jednotlivých odběratelů. Sestava bude obsahovat pole **Firma**, **Datum**, **Kusy**, **Nazev**, **Cena**, **Faktura**. Seskupené budou podle **Firmy** a dále podle **Názvu** zboží. Seřaďte záznamy podle **Data prodeje** vzestupně. Nastavené rozložení bude Odsazované. Název sestavy zadejte **Prodej podle odběratelů**. Upravte sestavu takto:
   1. Pozadí záhlaví **Firma** bude změněno na žlutou barvu. Nebude se u každé druhé firmy přepínat na jinou barvu.
   2. U pole **Nazev** změňte barvu písma na modrou.
   3. Zajistěte, aby záhlaví názvu zboží nebylo osamoceno na konci stránky.

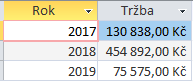
## Relace M:N – pokračovaní příkladu Grily

Databázi **Grily** budeme modifikovat tak, abychom nemuseli zapisovat neustále ceny zboží do tabulky **Prodeje**. Vytvoříme tedy další tabulku pomocí dotazu s názvem **Zbozi**, kde budou údaje o názvu zboží, ceně, popř. další podrobnosti. Tuto tabulku potom svážeme relací s tabulkou **Prodej**. Vznikne další relace **1:N** a následně se tím vytvoří relace označovaná jako **N:N**.

1. Vytvořte dotaz, který vytvoří novou tabulku s názvem **Zbozi**. V ní bude figurovat sloupeček **Nazev** a **Cena** z tabulky **Prodej** a bude obsahovat jedinečné hodnoty. Dotaz uložte jako **ZBO1-D**. Dotaz spusťte.
2. Přejmenujte sloupeček **Nazev** v tabulce **Zbozi** na **Druh** a doplňte do tabulky zcela nový sloupeček **Vlastnosti**.
3. Vložte do tabulky tyto údaje:



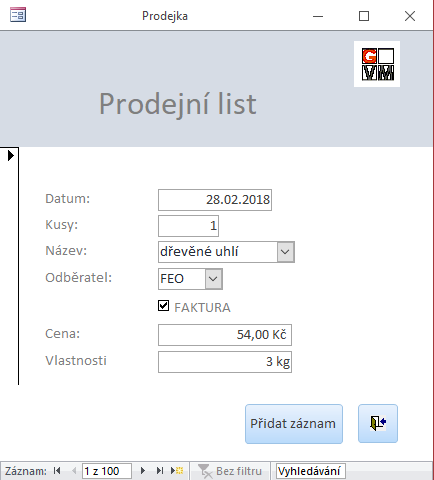
1. Na pole **Druh** nastavte primární klíč.
2. Zajistěte vytvoření rozbalovací nabídky u pole **Nazev** v tabulce **Prodej**. Zdrojem dat nabídky bude tabulka **Zbozi**.
3. Odstraňte sloupeček **Cena** z tabulky **Prodej**.
4. Upravte relaci mezi polem **Druh** z tabulky **Zbozi** a polem **Nazev** z tabulky **Prodej** tak, aby do tabulky **Prodej** nemohl být přidán žádný záznam se zbožím, který neexistuje v tabulce **Zbozi**. Zajistěte možnost případné změny názvu zboží.
5. Vytvořte dotaz, který zjistí součet tržeb v jednotlivých letech. Tržby budou ve formátu Kč, zobrazte dvě desetinná místa. Dotaz nazvěte **Trzby-D**.



1. Vytvořte křížový dotaz, který zjistí součet kusů u jednotlivých firem a jednotlivých výrobků. Dotaz nazvěte **Krizovy-D**.



1. Podle vzoru vytvořte formulář pro prodej zboží jednotlivým odběratelům. Pole **Cena** a **Vlastnosti** budou uzamčeny. Pozor na to, že některá pole jsou z tabulky **Zbozi**, jiná pole z tabulky **Prodej**. Do záhlaví formuláře vložte název **Prodejní list**. Do formuláře dále vložte logo naší školy. Formulář nazvěte **Prodejka**.



1. Pomocí formuláře vložte tento záznam do databáze a poté zkontrolujte, zda byl záznam přidán do tabulky Prodej.

