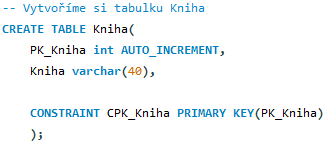
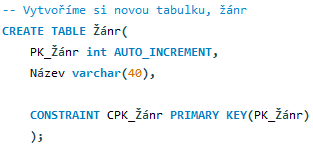
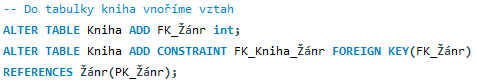
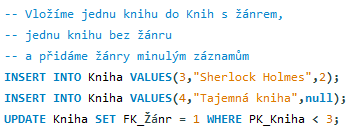
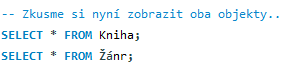
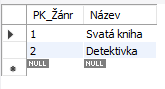
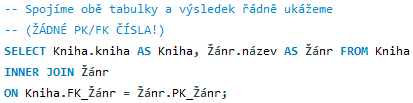
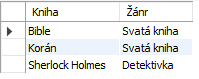
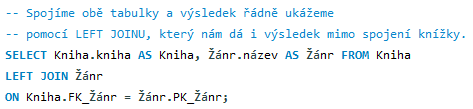
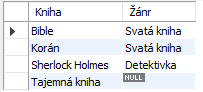
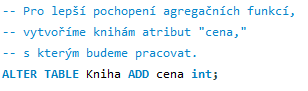
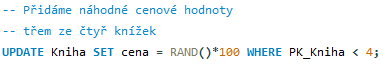
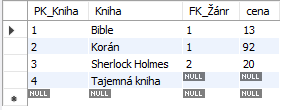
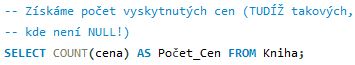
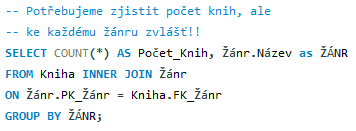
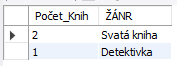
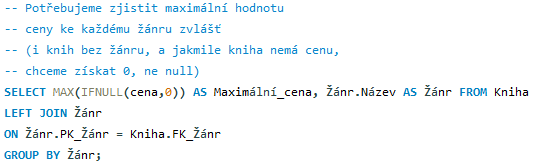
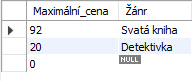
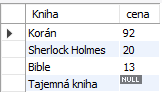
**12. Jazyk SQL – SELECT, VIEW (spojování tabulek, agregační funkce, seskupování záznamů)**

**Základní pojmy**

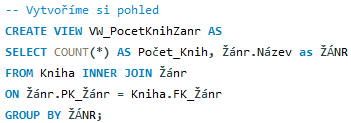
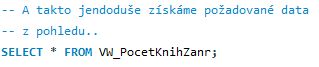
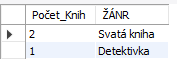
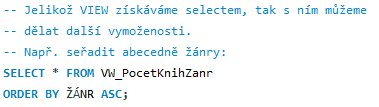
* Nějakým způsobem musíme **získávat data z databáze**
* Právě proto existují SQL příkaz SELECT a objekt VIEW

**A picture containing rectangle, screenshot, font, square

Description automatically generatedSELECT**

* Spadá buďto do DML, či do samotné skupiny **DQL – Data Query Language** *(Záleží na názoru a úhlu pohledu)*
* SELECT “tahá” data ze čitelných objektů (Tabulka, View), všelijak čitelné objekty spojuje a manipuluje s daty, abychom z nich získali INFORMACE, které jsou požadovány
* **Základní SELECT**
  + 
  + 
  +   
    
  + **Můžeme používat aliasy *[AS]* jak ke sloupcům, tak k TABULKÁM!**  
      
    
    - Tyto aliasy se nejen zobrazí ve výsledku SELECTU, ale také se poté používají dále v SELECTU *(Jako například ve WHERE klauzule apod.)*
* **Spojování objektů v SELECTU**
  + Typy spojení:  
    - **INNER JOIN** – Objekty spojíme mezi sebou a vrátíme pouze ty výsledky, které mají spojení
    - **LEFT JOIN** – Objekty spojíme mezi sebou a vrátíme výsledky, které mají spojení, i s výsledky které spojení nemají z levého objektu
    - **RIGHT JOIN** – Objekty spojíme mezi sebou a vrátíme výsledky, které mají spojení, i s výsledky které spojení nemají z pravého objektu
    - **FULL OUTER JOIN** – Vrátí jak spojené výsledky, tak i nespojené výsledky *(V MySQL není)*
      * *(OUTER JOINU můžeme v MYSQL dosáhnout LEFT JOINU a RIGHT JOINU objektů + jejich UNION)*
    - **CROSS JOIN** – Spojí všechny výsledky z prvního objektu s každým výsledkem druhého objektu
  + Asi teď víme, že mezi sebou mají tabulky vztahy pomocí svých klíčů
  + To nám ale moc neřekne, když vidíme například pouze FK ID žánru, ve kterém se Knížka nachází, nikoliv jeho název...
  + To je jeden z důvodů, proč můžeme objekty “seskupit”, a zobrazit tak člověkem čitelná data *(Či díky seskupení získat data pro další potřebné operace)*
  + 
  + 
  + 
  +   
      
    - *To, že má kniha nějaký ŽÁNR 1, nám moc neřekne, co?..*  
    
  +   
    
    - Super! Máme správně seskupené knížky a žánry.
    - Chybí nám tam každopádně naše knížka “Tajemná kniha,” jelikož jsme použili **INNER JOIN**, který vrací pouze spojené instance... Ale “Tajemná kniha” nemá žádný žánr, tak s čím chceme spojovat?!
  +   
    
* **Agregační funkce**
  + Dají výsledky do SKUPINY, ze které poté získá data, a vrátí jeden výsledek za jednu skupinu *(BEZ GROUP BY (skupení) vrátí vždy pouze jeden výsledek)*
  + Jelikož se s agregačními funkcemi většinou počítá/nachází největší a nejmenší, většina funguje převážně jen na číslech a datumech
  + Typy Agregačních funkcí:
    - **AVG ()** - Vrátí průměrnou hodnotu vybraného sloupce
    - **COUNT ()** - Vrátí počet vyskytujících se hodnot vybraného sloupce (BEZ NULL)
    - **MAX ()** - Vrátí maximální nalezenou hodnotu vybraného sloupce
    - **MIN ()** - Vrátí minimální hodnotu vybraného sloupce
    - **SUM ()** - Vrátí sumu (součet) celého vybraného sloupce
  + 
  +   
    
  +   
    
  +   
    
  +   
    
  +   
    
  + Všimněme si, že agregační funkce kompletně ignorují NULL hodnoty.
  + AVG, MAX a MIN pracují pouze s reálnými hodnotami
* **Seskupování záznamů**
  + Někdy jsou třeba data agregačních funkcí, ale rozděleně ve více skupinách.
  + Proto používáme seskupení **GROUP BY**, kde seskupíme výsledky agregačních funkcí
  +   
    
  + 
  + 
* **Řazení**
  + Můžeme výsledek také seřadit dle sloupců pomocí klauzule **ORDER BY**
    - **ASC** – Hodnoty budou od nejmenšího po největší
    - **DESC** – Hodnoty budou od největšího po nejmenší
  +   
    

**VIEW**

* Objekt, který v sobě **uchovává *(většinou)* složité SELECTY** pod nějakým **názvem** za účelem **jednoduchého spuštění SELECTU** a získání tak požadovaných dat
* Do pohledu **můžeme i INSERTOVAT data** *(přidají se pak do tabulky)*, když neobsahují agregační funkce
  + SELECT pro zjištění počtu knih pro každý žánr byl docela veliký... Představme si tento SELECT psát pokaždé, kdy tuto informaci budete potřebovat. Pojďme ho dát do VIEW a podívat se, jak jednoduché je jeho spuštění:
  + 
  +   
    
  +   
    
  + 