**mysql** -u root –p: Pripojeni k databazi.

CREATE database engeto: Vytvori databazi

SHOW databases: Ukaze database

SHOW tables: Ukaze tabulky.

**\c:** zrusi dotaz

**USE** “nazev”: Vybere databazi, se kterou budeme pracovat

**CREATE TABLE** “nazev”: Vytvoreni tabulky.

**PRIMARY KEY(id):**  integritní omezení, které nám říká, že id je primárním klíčem tabulky.

**DESCRIBE**: ukáže strukturu tabulky

ALTER TABLE customers ADD COLUMN testcol VARCHAR(255); Prida sloupec do tabulky.

DROP: Smazani.

ALTER TABLE customers DROP COLUMN testCol;

DROP TABLE …. : Smaze tabulku.

DROP DATABASE testdb; Smaze databazi.

INSERT INTO: pridani zaznamu do tabulky

INSERT INTO customers (firstName, lastName, email, address, city, state) VALUES('John', 'Smith', 'john@engeto.com', 'Cyrilska 7', 'Brno', 'Czech Republic');

UPDATE: upraveni, vlozeni novych dat.

UPDATE customers SET age = 23 WHERE id = 1;

DELETE FROM: Mazani.

DELETE FROM customers WHERE city = 'Bratislava';

DIV: Deleni se zaokrouhlenim.

%: Zbytek po deleni, MODULO. napr. 1024 % 50

**Logické operátory**

! NOT: Logický zápor. Vyhodnocuje 1, pokud je operand 0, a 0, pokud je operand nenulový. Dále aplikace na NOT NULL vrací NULL.

&& konjunkce: Logické AND. Vyhodnotí 1, pokud jsou všechny operandy nenulové a není mezi nimi NULL. Na 0, pokud je alespoň jeden operand 0.

|| disjunkce: Logické NEBO. Pokud oba operandy nejsou NULL a pokud je jakýkoli operand nenulový, výsledek je 1, jinak 0. U operandu NULL je výsledek 1, pokud je druhý operand nenulový, a jinak NULL. Pokud jsou oba operandy NULL, výsledkem je NULL.

XOR výlučná disjunkce: XOR je zkratka pro eXclusive OR. Vrací NULL, pokud je některý operand NULL. U operandů, které nejsou NULL, se vyhodnocuje na 1, pokud je lichý počet operandů nenulový, jinak se vrátí 0.

**Agregačních funkce**:

**COUNT():** vrátí počet řádků splňujících criteria, zjisteni poctu zaznamu

**MIN():** zjistí minimální (nejnižší) hodnotu z předložených řádků.

**MAX():** naopak zjistí maximální (nejvyšší) hodnotu z předložených řádků.

**SUM():** zajistí součet hodnot v daných řádcích.

**AVG():** vrátí průměrnou hodnotu v daných řádcích.

GROUP BY: Seskupy data podle urcitych vlastnosti.

SELECT dept, MAX(salary) FROM employees GROUP BY dept;

SELECT COUNT(\*) AS "Počet lidí v Engineering department" FROM employees WHERE dept = 'Engineering';

AS: vytvori nebo prejmenuje nazev

**CONCAT:** ma na vstupu libovolné množství argumentů a všechny je spojí do **jednoho řetězce**.

SELECT CONCAT('Enge', 'to', ' je ', 'super!');

SELECT CONCAT(firstName, ' ',lastName, ' (', email, ')') AS "Informace o zákazníku" FROM customers;

**GROUP\_CONCAT:** spojování **skupin řetězců**.

Separator “ ” – nastavime jakym znakem oddelit.

**CHAR\_LENGTH**: Vrati delku retezce.

**BIT\_LENGTH:** Vrati delku v bitech.

SELECT

BIT\_LENGTH('Úplně žluťoučký kůň pěl ďábelské ódy.') AS "UNICODE",

BIT\_LENGTH('Uplne zlutoucky kun pel dabelske ody.') AS "ASCII";

**SUBSTRING** (name, 1, 2): zadame odkud bere, kde zacina a kolik pismen. **PREFIX**-od zacatku

(name, -2, 2) SUFLIX: od konce

**STRCMP:** Rovnost retezcu

* **0**: řetězce jsou **totožné**.
* **-1**: řetězce nejsou stejné a **první z nich je abecedně dříve**.
* **1**: řetězce nejsou stejné a **druhý z nich je abecedně dříve**.

**LOCATE(substr,str)**: vrátí pozici začínajícího podřetězce a nulu, pokud se řetězec nevyskytuje.

SELECT LOCATE('cool', 'Engeto je cool.');

**LOCATE(substr,str,pos)**: dělá totéž, pouze začíná v řetězci hledat až od námi určené pozice dle parametru pos.

**LIKE** umožňuje vyhledávat v tabulce jenom podle části textu, takže nemusíme zadat celou hodnotu.

* \_ **podtržítko**: Reprezentuje jeden libovolný znak.
* SELECT \* FROM customers WHERE city LIKE "\_\_\_\_\_\_\_\_"; počet znaku
* % **procento**: Reprezentuje libovolný počet (0 až n) libovolných znaků.

SELECT \* FROM customers WHERE email LIKE "%gmail%";

**TRIM:** bili znak, mezera tabulator

**LTRIM**: Smaže všechny bílé znaky z levé strany vstupního řetězce.

**RTRIM**: Smaže všechny bílé znaky z pravé strany vstupního řetězce.

**TRIM**: Smaže všechny bílé znaky z obou stran vstupního řetězce. Tato funkce umí odstraňovat i jiné typy znaků za použití přepínačů **LEADING a TRAILING**.

SELECT TRIM(LEADING 'x' FROM 'xxxxEngeto'); vymaze ‚‘x‘

**QUOTE:** zrušení speciální funkce jednoduché uvozovky.

SELECT QUOTE("Let's write SQL!");

**JOIN:** Spojeni tabulek

**LEFT JOIN:** vezme vsechny hodnoty y leveho sloupce a je NULL nahradi nulou

**RIGHT JOIN:** Opak leveho

**UNION:** spoji tabulky, musi byt stejne(pismena nebo cisla)

**Modulo %11,** bankovni ucty a rodna cisla.

**Date\_format (odkud, ‘format’, jazyk)**

**date\_format**(cp.date\_to, '%e. %M %Y', 'cs\_CZ')

**HAVING** lepe spolupracuje s GROUP BY, dava se az za.