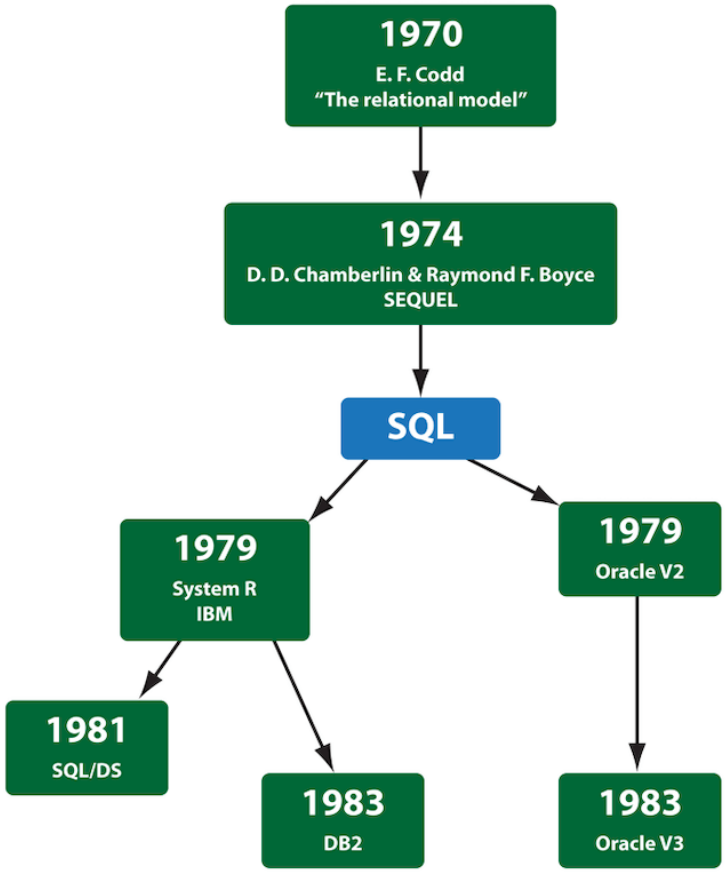
Databázové systémy

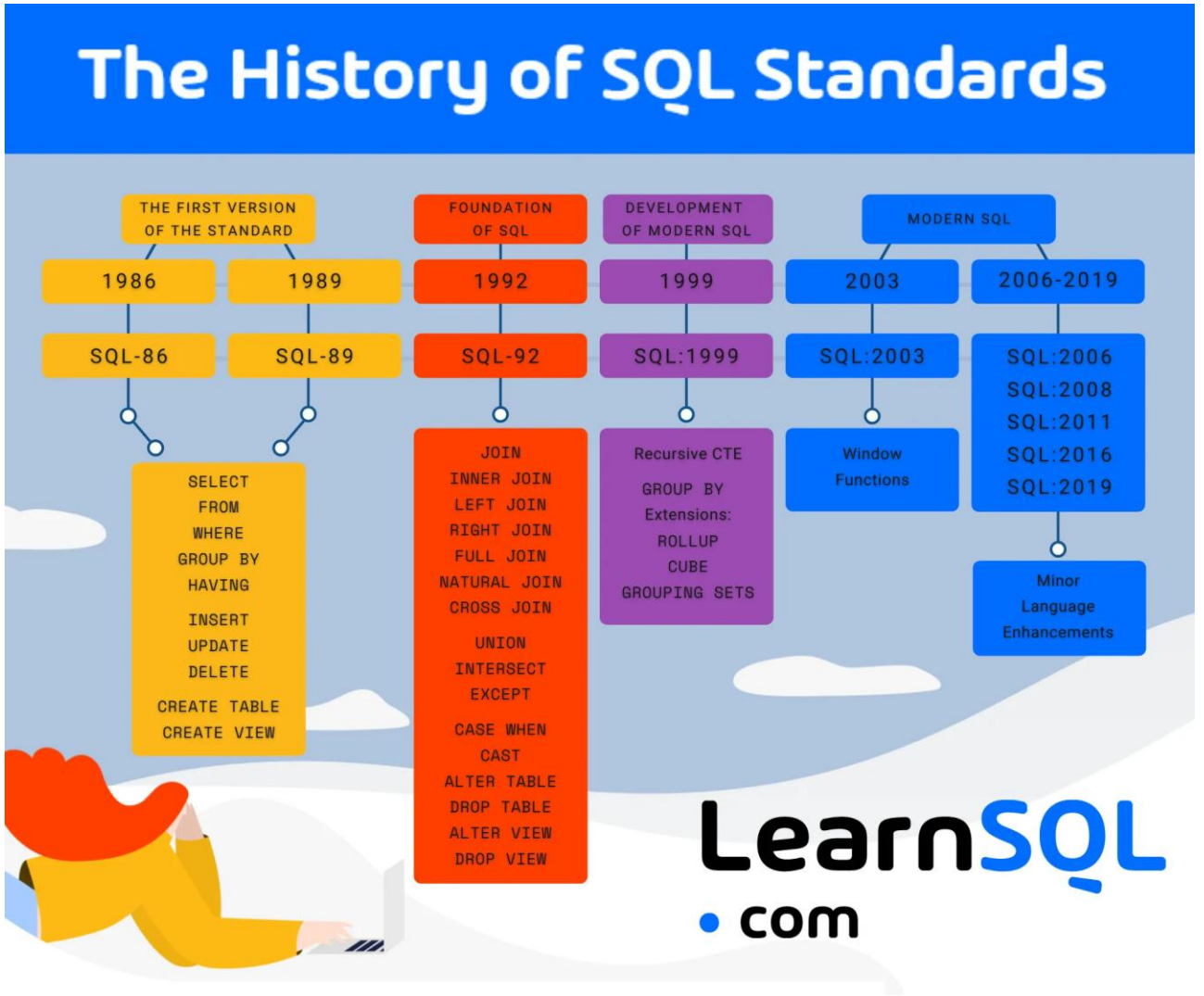
SQL

2026 Lukáš Miňo

SQL = Structured Query Language



https://clarusway.com/wp-content/uploads/2021/09/sql-history.png



https://learnsql.com/blog/history-of-sql-standards/history-of-sql-standards.webp

Výhody

$$\sigma_{id=5}(\pi_{meno,priezvisko}(Student)))$$

SELECT meno, priezvisko FROM Student WHERE id = 5

- **SQL** = Structured Query Language
- **RDBMS** = Relational Database Management System
- **Vek -** Jazyk vznikol už v 70. rokoch v IBM a stále je najpoužívanejší.
- **Matematický základ** - postavený na **relačnej algebre** (množinové operácie ako zjednotenie, prienik, rozdiel, kartézsky súčin) a **relačnom kalkule** (predikátová logika prvého rádu).
- **Štandardizácia:** ANSI a ISO, od SQL-86 po SQL:2023 (JSON, SQL/PGQ)
- **Deklaratívna paradigma**- what to do
  - vytvor databázu, vlož dáta, vyber dáta, odfiltruj dáta, zoraď dáta, aktualizuj dáta, vymaž dáta, zobraz tabuľky, ...
  - o procedurálne vykonanie, optimalizáciu a prístup k fyzickým súborom sa stará **Query Optimizer** v RDBMS
- Imperatívny paradigma - how to do things
  - premenné, podmienky, vetvenie, cykly, funkcie, výnimky, rekurzia, ...

- Predstavte si SQL ako čašníka v reštaurácii. Vy (používateľ) mu poviete objednávku z jedálneho lístka a čašník (SQL) zájde do kuchyne (databázy) a prinesie vám presne to jedlo, ktoré ste chceli. Nemusíte vedieť, ako kuchár varil.

Relačné vs nerelačné databázy

- **Relačné databázy (SQL):** Dáta sú usporiadané do prísnych **tabuliek.** Všetko má svoj jasný poriadok, pravidlá a vzťahy (relácie).
  - Oracle, MySQL, MS SQL Server, PostgreSQL, DB2, MariaDB, ...
- **Nerelačné databázy (NoSQL):** Dáta sú flexibilné, často uložené ako textové dokumenty (JSON). Používa sa tam, kde sa štruktúra dát neustále mení.
  - MongoDB, Redis.

- NoSQL je trendy, ale **SQL je základom biznisu**. NoSQL je trendy, ale **SQL je základom biznisu**. Viac ako 80 % firemných aplikácií beží na relačných databázach. Viac ako 80 % firemných aplikácií beží na relačných databázach.

Prečo je SQL stále populárne?

- **Nie je to len pre programátorov:** SQL dnes **povinne** používajú dátoví analytici, produktoví manažéri, testeri (QA), business analytici…
- **Univerzálnosť:** Ak sa naučíte syntax pre MySQL, bez problémov viete písať dopyty v PostgreSQL alebo Oracle. Princíp je na 95 % rovnaký.
- **Dopyt po dátach:** Žijeme v dobe dátového boomu. Firmy ako Netflix, Spotify, Instagram, či Uber generujú terabajty dát za sekundu. Všetky tieto dáta sú niekde uložené a niekto ich musí analyzovať.

Úloha SQL v modernej softvérovej architektúre

- **Perzistentná vrstva:** V trojvrstvovej architektúre (Presentation -> Application -> Data) tvorí SQL databáza kritický základ, na ktorý sa viaže aplikačná logika.
- **Abstrakcia cez ORM vs. Native SQL:** Moderné frameworky využívajú ORM (*Object-Relational Mapping* ako Hibernate, Entity Framework, Prisma). Prečo teda študovať SQL?
  - **Výkonnostné limity ORM:** Pri komplexných analytických dopytoch (OLAP) generujú ORM **neoptimálny kód** (**problém N+1 dopytov**).
  - **Optimalizácia a Profiling:** Bez hlbokej znalosti SQL vývojár nedokáže čítať exekučný plán (EXPLAIN, EXPLAIN ANALYZE) a diagnostikovať úzke hrdlá (Bottlenecks) v databáze.
- **Dátové inžinierstvo a OLAP:** SQL je dominantným jazykom v dátových skladoch (Data Warehouses) ako Snowflake, BigQuery, ClickHouse na transformáciu dát (procesy **ETL** (Extract, Transform, Load) / **ELT** (Extract, Load, Transform)).

SQL syntax

Syntaktická flexibilita vs. Profesionálne konvencie

- **Case Sensitivity (Citlivosť na veľkosť písmen):**
  - **Kľúčové slová:** SQL je z hľadiska kľúčových slov **case-insensitive** (select je to isté ako SELECT). Štandardom je však písať **kľúčové slová striktne VEĽKÝMI PÍSMENAMI** (SELECT, FROM, WHERE), aby sa vizuálne oddelili od logiky biznisu.
  - **Identifikátory (názvy tabuliek a stĺpcov):** Citlivosť závisí od operačného systému a konfigurácie RDBMS (napr. Linux vs. Windows pri MySQL). Konvenciou je písať názvy tabuliek a stĺpcov **malými písmenami**.
    - **snake_case***:* Viacslovné názvy sa oddeľujú podčiarkovníkom (napr. datum_narodenia, objednavky_polozky). Vyhýbajte sa CamelCase.

Syntaktická flexibilita vs. Profesionálne konvencie

- **Ukončovanie príkazov (Bodkočiarka ;):**
  - V mnohých databázach (napr. MySQL, PostgreSQL) je bodkočiatka na konci samostatného dopytu voliteľná, ak ide o jediný príkaz.
  - **Best Practice:** Vždy explicitne ukončujte každý príkaz bodkočiarkou. Je to striktne vyžadované pri spájaní viacerých dopytov do skriptov, pri písaní uložených procedúr (Stored Procedures) či CTE (Common Table Expressions).
- **Dokumentácia kódu (Komentáre):**
  - **Jednoriadkový komentár:** Začína dvoma pomlčkami `-- Tento dopyt filtruje aktívnych študentov`. (Pozor: v niektorých dialektoch musí po `--` nasledovať medzera). Alternatívne sa v niektorých DB používa mriežka `#`.
  - **Viacriadkový / Blokový komentár:** Ohraničený pomocou `/* ... */`, identicky ako v jazykoch C++ alebo Java. Používa sa na rozsiahlejšiu dokumentáciu komplexných dopytov.
- **Formátovanie a Odsadzovanie (White Spaces):**
  - Pre SQL sú tabulátory, medzery a nové riadky irelevantné (dopyt môžete napísať do jedného nekonečného riadku).
  - **Pravidlo čitateľnosti:** Každú hlavnú klauzulu (`SELECT`, `FROM`, `WHERE`, `GROUP BY`) píšte na **nový riadok** a zoznam stĺpcov alebo podmienok vizuálne odsadzujte.

Syntaktická flexibilita
vs. Profesionálne konvencie

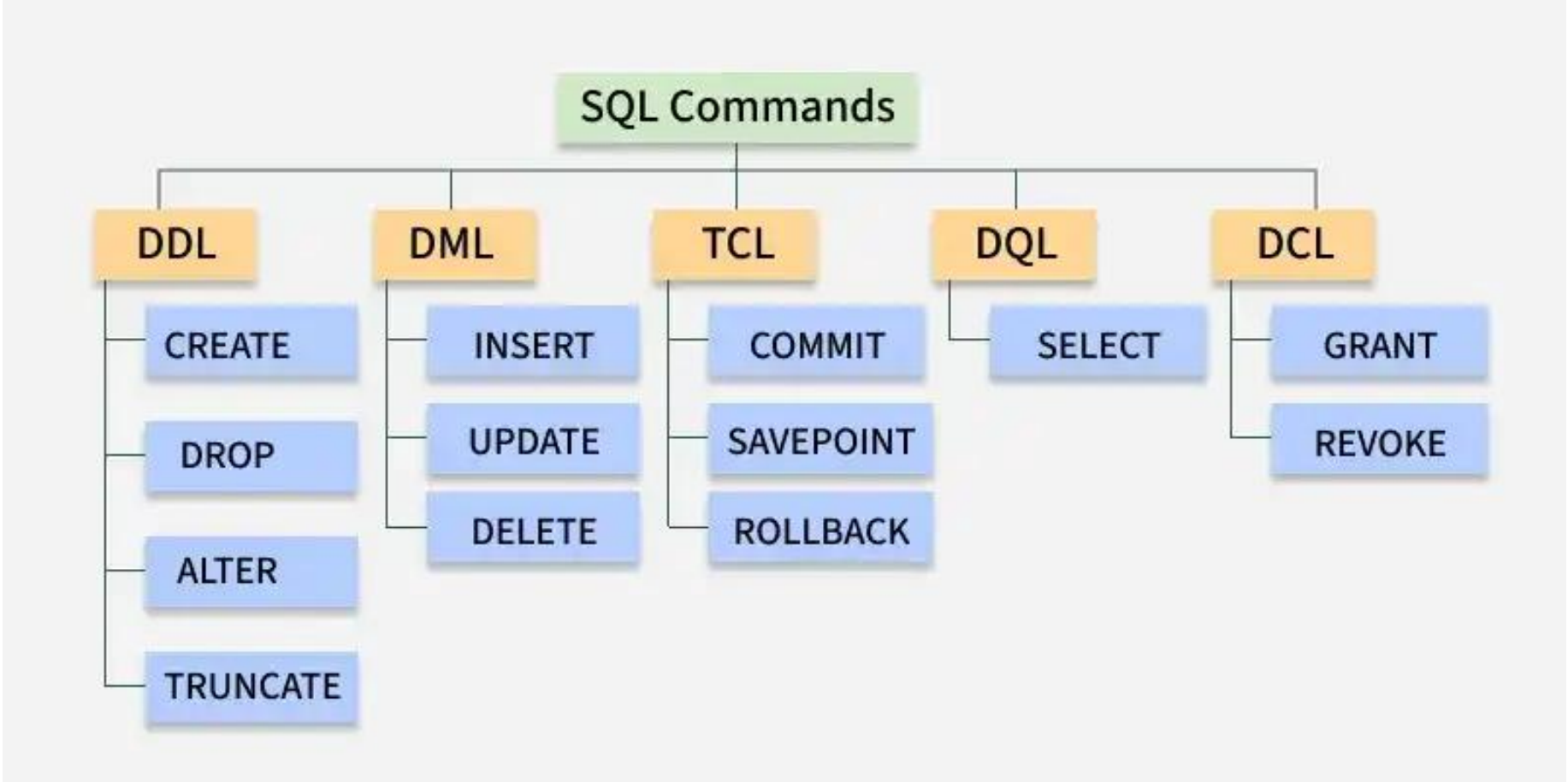
- **Anti-pattern (Syntakticky správne, ale nečitateľné):**

```
select meno, priezvisko from osoba where pohlavie = 'm' and otec is not null
```

- **Profesionálny zápis (Čitateľný a udržateľný):**

```
SELECT
        meno,
        priezvisko
FROM
        osoba
WHERE
        pohlavie = 'm’ AND otec IS NOT NULL;
```

SQL príkazy



https://media.geeksforgeeks.org/wp-content/uploads/20260522121146555616/commands.webp

DDL = Data Definition Language

Príkaz	Popis	Syntax
CREATE	Vytvorí databázu alebo jej objekty (tabuľku, index, funkciu, pohľady, uloženú procedúru a triggre/spúšťače).	`CREATE TABLE table_name (column1 data_type, column2 data_type, ...);`
DROP	Vymaže objekty z databázy (odstráni ich štruktúru aj dáta).	`DROP TABLE table_name;`
ALTER	Mení a upravuje štruktúru existujúcej databázy alebo tabuľky.	`ALTER TABLE table_name ADD COLUMN column_name data_type;`
TRUNCATE	Odstráni všetky záznamy z tabuľky vrátane uvoľnenia celého priestoru, ktorý bol pre tieto záznamy vyhradený.	`TRUNCATE TABLE table_name;`
COMMENT	Pridá komentáre do dátového slovníka (data dictionary).	`COMMENT ON TABLE table_name IS 'comment_text';`
RENAME	Premenuje existujúci objekt v databáze.	`RENAME TABLE old_table_name TO new_table_name;`

```sql
DROP DATABASE IF EXISTS OsobaVztah;
CREATE DATABASE OsobaVztah;
USE OsobaVztah;

CREATE TABLE osoba (
   id INT NOT NULL PRIMARY KEY,
   meno VARCHAR(10) NOT NULL,
   priezvisko VARCHAR(20) NOT NULL,
   rodne_priezvisko VARCHAR(20),
   dat_nar DATE NOT NULL,
   dat_smrti DATE,
   pohlavie CHAR(1) NOT NULL CHECK (pohlavie IN ('m','z')),
   vyska DEC(4,1) CHECK (vyska BETWEEN 30.0 AND 250.0),
   vaha DECIMAL(4,1),     -- To iste ako DEC(4,1)
   otec INT ,
   matka INT ,
   FOREIGN KEY (otec) REFERENCES osoba(id),
   FOREIGN KEY (matka) REFERENCES osoba(id)   -- cudzi kluc s menom
);
ALTER TABLE osoba ADD email VARCHAR(50);
TRUNCATE TABLE osoba;
CREATE USER 'adam’@’localhost' IDENTIFIED BY 'heslo';
```

DDL = Data Definition Language

- CREATE/ALTER/DROP
  - PROCEDURE
  - FUNCTION
  - TRIGGER
  - VIEW
  - INDEX
  - ...

- Viac sa dozviete v letnom semestri ;-)

Dátové typy

- **Numerické dátové typy**
  - Presné numerické – TINYINT, SMALLINT, MEDIUMINT, INTEGER=INT, BIGINT, DECIMAL(precision, scale)=DEC(precision, scale), ...
    - **precision** - je maximálny počet číslic (presnosť)
    - **scale** - označuje počet číslic za desatinnou čiarkou
  - Približné numerické - FLOAT, DOUBLE
- **Reťazcové dátové typy**
  - CHAR(size), VARCHAR(size)
  - BLOB, TEXT, ...
  - ENUM, SET
- **Dátumové a časové dátové typy**
  - DATE, TIME, DATETIME, TIMESTAMP, YEAR, ...
- Iné
  - JSON, XML, POINT, POLYGON, ...

Sémantika hodnoty NULL

- **Definícia:** NULL reprezentuje **absenciu hodnoty**, neznámu hodnotu a
- **Príklad z praxe:** Ak študent nemá zadané telefonne_cislo, hodnota je NULL (neznamená to, že jeho číslo je 0).
- **Vplyv na integritu:** Atribúty môžu mať obmedzenie NOT NULL, ktoré vynucuje, aby bola položka pri zápise vždy vyplnená.

Trojhodnotová logika (3VL – Three-Valued Logic)

- Prítomnosť NULL mení klasickú booleovskú logiku (True/False) na trojhodnotovú. Akýkoľvek aritmetický alebo porovnávací výraz, v ktorom figuruje NULL, vracia výsledok **UNKNOWN** (Neznáme).

∧	T	U	F
T	T	U	F
U	U	U	F
F	F	F	F

∨	T	U	F
T	T	T	T
U	T	U	U
F	T	U	F

¬	
T	F
U	U
F	T

https://stevengong.co/attachments/Screenshot-2023-06-24-at-8.32.49-PM.png

DML = Data Manipulation Language

Príkaz	Popis	Syntax
INSERT	Vloží dáta do tabuľky	`INSERT INTO table_name (column1, column2, ...) VALUES (value1, value2, ...);`
UPDATE	Aktualizuje existujúce dáta v tabuľke	`UPDATE table_name SET column1 = value1, column2 = value2 WHERE condition;`
DELETE	Vymaže záznamy z databázovej tabuľky	`DELETE FROM table_name WHERE condition;`

```
INSERT Osoba
   VALUES (4, 'Nasťa', 'Kováčová', 'Rostová', '1928-02-05', '1965-03-11', 'z', 155.0, 99, NULL, NULL);

INSERT INTO Osoba (id, priezvisko, meno, rodne_priezvisko, dat_nar, dat_smrti, pohlavie, vyska, vaha, otec, matka )
   VALUES(5,'Urban' , 'Jozef', NULL, '1922-10-19', NULL, 'm', 199.5, Null, NULL, NULL); -- meno vs. priezvisko

INSERT INTO Osoba (id,meno, priezvisko, rodne_priezvisko, dat_nar, dat_smrti, pohlavie, vyska, vaha, otec, matka )
   VALUES
    (6, 'Mária', 'Urbanová', 'Nováková', '1937-12-08', NULL, 'z', 172.5, 57.5, 1, 2 ),
    (7, 'Patrik', 'Novák', 'Novák', '1945-06-19', NULL, 'm', 182.5, 89.5, 1, 2 ),
    (8, 'Patrícia','Nováková', 'Halušková', '1952-01-08', NULL, 'z', 143.5, 35, NULL, NULL),
    (9, 'Michal', 'Kováč', 'Kováč', '1942-04-10', NULL, 'm', 167.0, 88, 3, 2 );

INSERT Osoba
VALUES
   (10,'Roman', 'Kováč', 'Kováč', '1948-05-20', NULL, 'm', 179.5, 78.5, 3, 4 ), -- aj tak sa da :)
   (11,'Peter', 'Horváth', 'Horváth', '1959.7.2', '2000.12.31','m', 193.0, 110.5,NULL, NULL);

UPDATE osoba SET meno = 'Dáša', rodne_priezvisko = priezvisko, priezvisko = 'Dášová' WHERE id = 24;
DELETE FROM osoba WHERE id = 24;
```

TCL - Transaction Control Language

Príkaz	Popis	Syntax
BEGIN TRANSACTION	Spustí novú transakciu	`BEGIN TRANSACTION [transaction_name];`
COMMIT	Uloží všetky zmeny vykonané počas transakcie	`COMMIT;`
ROLLBACK	Vráti späť (zruší) všetky zmeny vykonané počas transakcie	`ROLLBACK;`
SAVEPOINT	Vytvorí bod uloženia (savepoint) v rámci aktuálnej transakcie	`SAVEPOINT savepoint_name;`

```
BEGIN TRANSACTION;
SAVEPOINT before_update;
UPDATE vztah SET do = current_date() WHERE id_on = 1;
DELETE osoba WHERE id = 1;
ROLLBACK TO SAVEPOINT before_update;
COMMIT;
```

```
SELECT * FROM osoba;
```

id	meno	priezvisko	rodne_priezvisko	dat_nar	dat_smrti	pohlavie	vyska	vaha	otec	matka
1	Adam	Prvý	NULL	1918-05-11	1968-10-01	m	180.0	80.0	NULL	NULL
2	Eva	Prvá	Druhá	1919-01-09	1988-07-22	z	160.0	60.0	NULL	NULL
3	Zoly	Múdry	NULL	1918-04-07	1990-09-23	m	175.5	75.0	NULL	NULL
4	Nasa	Kováèová	Rostová	1928-02-05	1965-03-11	z	155.0	99.0	NULL	NULL
5	Jozef	Urban	NULL	1922-10-19	NULL	m	199.5	NULL	NULL	NULL
6	Mária	Urbanová	Nováková	1937-12-08	NULL	z	172.5	57.5	1	2
7	Patrik	Novák	Novák	1945-06-19	NULL	m	182.5	89.5	1	2
8	Patrícia	Nováková	Halušková	1952-01-08	NULL	z	143.5	35.0	NULL	NULL
9	Michal	Kováè	Kováè	1942-04-10	NULL	m	167.0	88.0	3	2
10	Roman	Kováè	Kováè	1948-05-20	NULL	m	179.5	78.5	3	4
11	Peter	Horváth	Horváth	1959-07-02	2000-12-31	m	193.0	110.5	NULL	NULL
12	Lucia	Horváthová	Urbanová	1959-01-13	NULL	z	156.5	45.5	5	6
13	Urban	Urban	Urban	1957-03-31	NULL	m	138.2	24.5	5	6
14	DÁŠa	Nováková	Novaková	1970-07-17	NULL	z	167.0	55.0	7	8
15	Viera	Silná	Novaková	1973-02-13	NULL	z	169.5	63.0	7	8
16	Vladimír	Silný	Silný	1974-08-01	2002-12-04	m	175.5	73.0	NULL	NULL
17	Milena	Slabá	Slabá	1979-09-14	NULL	z	164.0	64.0	NULL	NULL
18	Ján	Horváth	Horváth	1982-01-16	NULL	m	159.5	65.5	11	12
19	Zuzana	Silná	Silná	2002-03-01	NULL	z	158.5	60.0	16	15
20	Zuzana	Slabá	Slabá	1999-12-16	NULL	z	171.5	54.5	16	17
21	Zuzana	Pravá	Pravá	1990-11-26	NULL	z	170.5	60.5	16	17
22	Zuzana	¼avá	¼avá	1931-01-14	NULL	z	195.5	58.5	16	17
23	Zuzana	Stredná	Stredná	1945-04-08	NULL	z	150.5	87.0	16	17
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
SELECT

        meno,

        priezvisko AS 'Priezvisko‘,

        dat_nar 'Dátum narodenia‘

FROM

        osoba

WHERE

        pohlavie = 'z' AND priezvisko <> rodne_priezvisko AND dat_smrti IS NULL
        AND dat_nar < '1990-01-01' AND vyska > 160

ORDER BY priezvisko ASC, 1 DESC, 'Dátum narodenia';
```

meno	Priezvisko	Dátum narodenia
Viera	Silná	1973-02-13
Mária	Urbanová	1937-12-08

DQL - Data Query Language

Príkaz	Popis	Syntax
SELECT	Používa sa na získanie (výber) dát z databázy	`SELECT column1, column2, ... FROM table_name WHERE condition;`
FROM	Označuje tabuľku (alebo tabuľky), z ktorej sa majú dáta získať	`SELECT column1 FROM table_name;`
WHERE	Filtruje riadky pred akýmkoľvek zoskupovaním alebo agregáciou	`SELECT column1 FROM table_name WHERE condition;`
GROUP BY	Zoskupuje riadky, ktoré majú rovnaké hodnoty v špecifikovaných stĺpcoch	`SELECT column1, AVG_FUNCTION(column2) FROM table_name GROUP BY column1;`
HAVING	Filtruje výsledky príkazu GROUP BY	`SELECT column1, AVG_FUNCTION(column2) FROM table_name GROUP BY column1 HAVING condition;`
DISTINCT	Odstraňuje duplicitné riadky z množiny výsledkov	`SELECT DISTINCT column1, column2, ... FROM table_name;`
ORDER BY	Zoraďuje množinu výsledkov podľa jedného alebo viacerých stĺpcov	`SELECT column1 FROM table_name ORDER BY column1 [ASC | DESC];`
LIMIT	Používa sa na obmedzenie počtu vrátených riadkov v dopyte SELECT (bežne podporované v MySQL a PostgreSQL)	`SELECT * FROM table_name LIMIT number;`

```
SELECT * FROM osoba;
```

id	meno	priezvisko	rodne_priezvisko	dat_nar	dat_smrti	pohlavie	vyska	vaha	otec	matka
1	Adam	Prvý	NULL	1918-05-11	1968-10-01	m	180.0	80.0	NULL	NULL
2	Eva	Prvá	Druhá	1919-01-09	1988-07-22	z	160.0	60.0	NULL	NULL
3	Zoly	Múdry	NULL	1918-04-07	1990-09-23	m	175.5	75.0	NULL	NULL
4	Nasa	Kováèová	Rostová	1928-02-05	1965-03-11	z	155.0	99.0	NULL	NULL
5	Jozef	Urban	NULL	1922-10-19	NULL	m	199.5	NULL	NULL	NULL
6	Mária	Urbanová	Nováková	1937-12-08	NULL	z	172.5	57.5	1	2
7	Patrik	Novák	Novák	1945-06-19	NULL	m	182.5	89.5	1	2
8	Patrícia	Nováková	Halušková	1952-01-08	NULL	z	143.5	35.0	NULL	NULL
9	Michal	Kováè	Kováè	1942-04-10	NULL	m	167.0	88.0	3	2
10	Roman	Kováè	Kováè	1948-05-20	NULL	m	179.5	78.5	3	4
11	Peter	Horváth	Horváth	1959-07-02	2000-12-31	m	193.0	110.5	NULL	NULL
12	Lucia	Horváthová	Urbanová	1959-01-13	NULL	z	156.5	45.5	5	6
13	Urban	Urban	Urban	1957-03-31	NULL	m	138.2	24.5	5	6
14	DÁŠa	Nováková	Novaková	1970-07-17	NULL	z	167.0	55.0	7	8
15	Viera	Silná	Novaková	1973-02-13	NULL	z	169.5	63.0	7	8
16	Vladimír	Silný	Silný	1974-08-01	2002-12-04	m	175.5	73.0	NULL	NULL
17	Milena	Slabá	Slabá	1979-09-14	NULL	z	164.0	64.0	NULL	NULL
18	Ján	Horváth	Horváth	1982-01-16	NULL	m	159.5	65.5	11	12
19	Zuzana	Silná	Silná	2002-03-01	NULL	z	158.5	60.0	16	15
20	Zuzana	Slabá	Slabá	1999-12-16	NULL	z	171.5	54.5	16	17
21	Zuzana	Pravá	Pravá	1990-11-26	NULL	z	170.5	60.5	16	17
22	Zuzana	¼avá	¼avá	1931-01-14	NULL	z	195.5	58.5	16	17
23	Zuzana	Stredná	Stredná	1945-04-08	NULL	z	150.5	87.0	16	17
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
SELECT

        meno,

        priezvisko AS 'Priezvisko‘,

        dat_nar 'Dátum narodenia‘

FROM

        osoba

WHERE

        pohlavie = 'z' AND priezvisko <> rodne_priezvisko AND dat_smrti IS NULL
        AND dat_nar < '1990-01-01' AND vyska > 160

ORDER BY priezvisko ASC, 1 DESC, 'Dátum narodenia';
```

meno	Priezvisko	Dátum narodenia
Viera	Silná	1973-02-13
Mária	Urbanová	1937-12-08

DCL - Data Control Language

Príkaz	Popis	Syntax
GRANT	Prideľuje nové oprávnenia používateľskému účtu, čím mu umožňuje prístup ku konkrétnym databázovým objektom, akciám alebo funkciám.	`GRANT privilege_type [(column_list)] ON [object_type] object_name TO user [WITH GRANT OPTION];`
REVOKE	Odoberá predtým pridelené oprávnenia používateľskému účtu, čím ruší jeho prístup k určitým databázovým objektom alebo akciám.	`REVOKE [GRANT OPTION FOR] privilege_type [(column_list)] ON [object_type] object_name FROM user [CASCADE];`

```
GRANT SELECT, INSERT ON osoba TO 'adam’@’localhost';
FLUSH PRIVILEGES;
REVOKE INSERT ON osoba FROM 'adam’@’localhost';
```

```
GRANT ALL PRIVILEGES ON *.* TO 'adam'@'%’
WITH GRANT OPTION;
FLUSH PRIVILEGES;
```