Základy jazyka SQL

(Structured Query Language)

- vyvinula IBM začiatkom 70-tych rokov
- je to deklaratívny jazyk (popisuje **čo** urobiť, nie **ako**)
- je súčasťou veľkých databázových systémov (Informix, Oracle, DB2, dBASE IV)
- každý príkaz z jazyka SQL končí bodkočiarkou
- pracuje s relačnou databázou.

1 Vytvorenie tabuľky

CREATE TABLE <meno tabuľky>
(<meno stĺpca> <typ>(<dĺžka>)
[,<meno stĺpca> <typ>(<dĺžka>)]…);

- mená objektov databázy (tabuľky, stĺpce, ...) môžu obsahovať písmená, číslice a podčiarnik; musia začínať písmenom
- dĺžka identifikátorov je maximálne 30 znakov
- ako identifikátor sa nesmú použiť kľúčové slová

- možnosti pre <typ>: NUMBER, CHAR, DATE, WARCHAR2 (má väčšiu dĺžku reťazcov); ORACLE nepodporuje WARCHAR
- ak žiadame, aby daný stĺpec mal vždy hodnotu, za definíciu stĺpca vložíme klauzulu **NOT NULL**

  napr.
  CREATE TABLE Podnik (Evcis NUMBER (4) NOT NULL,
  .....;

- výpis všetkých tabuliek, ktoré sme vytvorili
  SELECT table_name FROM tabs;

Vkladanie dát do tabuľky

```
INSERT INTO <meno tabuľky> VALUES (
    <zoznam hodnôt>);
```

- jednotlivé hodnoty musia byť uvedené v takom poradí ako sú prvky tabuľky
- pri vkladaní do tabuľky môžeme použiť aj hodnotu **NULL**, ktorá predstavuje prázdnu hodnotu
- pomocou príkazu INSERT sa vkladá jeden nový záznam do tabuľky
- hodnoty typu CHAR sa dávajú do apostrofov
- pri vkladaní dátumu možno ako oddeľovač použiť pomlčku, lomítko alebo bodku;

Príklad: Vytvorte tabuľku popisujúcu pracovníkov inštitúcie

```
CREATE TABLE Podnik (
    Ev_cislo NUMBER(5) NOT NULL,
    Meno CHAR(25),
    Mesto CHAR(20),
    Ulica CHAR(15),
    Narodeny DATE,
    Mzda NUMBER(5),
    Poc_deti NUMBER(2),
    Veduci CHAR(25));

INSERT INTO Podnik
    VALUES (3241, 'Maly Jan', 'Kosice', 'Ruzova 15',
    '24-oct-1963', 16200, 3, 'Horny Ivan');
```

2 Manipulácia s tabuľkami

Pre formuláciu otázky v SQL slúži jediný príkaz SELECT, ktorý je základnou konštrukciou jazyka

2.1 Hľadanie v tabuľkách

```
SELECT <zoznam stĺpcov v projekcii>/*
    FROM <meno tabuľky>
        [WHERE <podmienka selekcie>];
```

***** - znamená výpis všetkých stĺpcov tabuľky.

Príklad: Vypíšte všetky mená a mestá z tabuľky Podnik

```
SELECT Meno, Mesto FROM Podnik;
```

Príklad: Vypíšte z tabuľky Podnik mená všetky všetkých košičanov

```
SELECT Meno FROM Podnik
    WHERE Mesto='Košice';
```

- systém stĺpcov nie je case sensitive, t.j. mená stĺpcov môžeme písať akýmikoľvek písmenami
- vo výpise sa mená stĺpcov vždy zobrazia veľkými písmenami
- SELECT vypisuje všetky hodnoty požadovaných stĺpcov – aj keď sú **rovnaké**
- ak požadujeme iba rôzne hodnoty použijeme klauzulu **DISTINCT**

Príklad: Zistite, z akých rôznych miest pracovníci podniku

```
SELECT DISTINCT Mesto FROM Podnik;
```

- pre zápis podmienky selekcie v príkaze WHERE je možné použiť:

  - **relačné operátory**

    = rovná sa

    pri porovnávaní reťazcov musíme vypísať celý reťazec, nie sú povolené žolíky

Príklad: Vypíšte mzdu Jána Malého

```
SELECT Meno, Mzda FROM Podnik
        WHERE Meno = 'Maly Jan';
```

Ďalšie:

!= alebo <> nerovná sa

> väčší

>= väčší alebo rovný

<= menší alebo rovný

< menší

BETWEEN

- testuje príslušnosť prvku k vybranému **uzavretému** intervalu

- BETWEEN <dolná hranica> AND <horná hranica>

Príklad: Vyhľadajte všetkých pracovníkov, ktorých mzda je medzi 10000 a 20000 Sk

```
SELECT Meno, Mzda FROM Podnik
    WHERE Mzda BETWEEN 10000 AND 20000;
```

IN

- testuje príslušnosť prvku do množiny
- syntax: IN (<zoznam prvkov množiny>)

Príklad: Vypíšte mená pracovníkov, ktorý majú 1 alebo 3 deti

```
SELECT Meno, Poc_deti FROM Podnik
        WHERE Poc_deti IN (1,3);
```

LIKE

- porovnáva reťazec s reťazcom, v ktorom sú použité žolíky
- žolíky
  podčiarnik **_** - nahradzuje **jeden** ľub. znak
  percento **%** - nahradzuje ľub. **skupinu** znakov

Príklad: Vypíšte všetky priezviská pracovníčok ukončené na – ová

```
SELECT Meno FROM Podnik
        WHERE Meno LIKE '%ova %';
```

➢ **logické operátory**
AND - logický súčin
OR - logický súčet
NOT - logická negácia

➢ **množinové operátory** (tieto Oracle ho nepodporuje)
INTERSECTION - prienik (je to isté ako AND)
UNION - zjednotenie (je to isté ako OR
MINUS - rozdiel (je to isté ako NOT AND)

➢ **konštanty a mená stĺpcov**

2.2 Usporiadanie riadkov v tabuľke

- záznamy z tabuliek sú vypisované v poradí v akom do tabuľky boli vložené

- ak chceme zmeniť ich poradie podľa hodnôt niektorého kľúča, použijeme klauzulu

ORDER BY <meno kľúča/zoznam kľúčov>

V príkaze SELECT

- **implicitne** sa výpis usporiada podľa hodnoty kľúča **vzostupne** (ASC)

- ak chceme opačné usporiadanie - **zostupné**, použijeme za menom kľúča klauzulu DESC

Príklad: Usporiadajte pracovníkov podľa výšky mzdy zostupne a v prípade, že majú rovnakú mzdu usporiadajte ich podľa abecedy

```
SELECT Meno, Mzda
    FROM Podnik
        ORDER BY Mzda DESC, Meno;
```

- pri použití viacerých kľúčov sa rešpektujú kľúče v takom poradí, v akom sú uvedené

- ak kľúčový stĺpec obsahuje prázdne hodnoty, záznamy s týmito hodnotami sa vypisujú ako prvé

2.3 Spojenie viacerých tabuliek

- príkazom SELECT, kde v klauzule FROM sa uvedú názvy spájaných tabuliek a v klauzule WHERE podmienka ich spojenia

  SELECT <zoznam stĺpcov>/*
  FROM <zoznam tabuliek>
  WHERE <podmienka spojenia>;

- ak spájané tabuľky majú rovnaké mená stĺpcov, musí sa použiť **kvalifikácia** stĺpcov: <meno tabuľky>.<meno stĺpca>

Príklad: Predpokladajme, že máme tabuľku Fond takejto štruktúry: Fond(Ev_cislo, Meno, Vyska). Nájdite adresu z tabuľky Podnik a výšku príspevku na fond z tabuľky Fond pre Jána Malého.

```
SELECT Podnik.Ev_cislo, Podnik.Meno, Fond.Vyska
    FROM Podnik, Fond
        WHERE Podnik.Ev_cislo = Fond.Ev_cislo AND
              Podnik.Meno = 'Maly Jan';
```

6. Modifikácia tabuľky

Pod **modifikáciou** tabuľky rozumieme :

- aktualizovanie prvkov tabuľky príkazom **UPDATE**
- zrušenie riadkov tabuľky príkazom **DELETE**
- pridanie stĺpcov tabuľky príkazom **ALTER TABLE ..... ADD**
- zmena rozmeru tabuľky príkazom **ALTER TABLE ..... MODIFY**
- vymazanie celej tabuľky príkazom **DROP**

➢ Aktualizovanie prvkov tabuľky

UPDATE <meno tabuľky>
SET <meno stĺpca> = <výraz1>
[, <meno stĺpca> = <výraz2>] ...
[WHERE <podmienka>];

- **Príklad**: Zvýšte mzdu všetkých odborných asistentov o tisíc korún.

```
UPDATE Fakulta
    SET Mzda = Mzda + 1000
        WHERE Funkcia = 'Odb_as';
```

➢ Zrušenie riadkov tabuľky

DELETE FROM <meno tabuľky>
[WHERE <podmienka>];

- **Príklad**: Zrušte záznam o pracovníkovi s menom Malý Ján

```
DELETE FROM Podnik
    WHERE Meno = 'Maly Jan';
```

➢ Pridanie nových stĺpcov

ALTER TABLE <meno tabuľky>
ADD (<meno stĺpca> <typ>
[, <meno stĺpca> <typ>] ...);

stĺpce sa pridávajú k tabuľke **sprava**

- pre odstránenie stĺpca z tabuľky v SQL **neexistuje** príkaz
- **riešenie**: nadefinuje sa nová tabuľka bez nežiadúcich stĺpcov a prekopírujú sa potrebné hodnoty z pôvodnej tabuľky príkazom **INSERT**

- **Príklad**: Prekopírujte tabuľku Podnik do tabuľky Podnik2.

```
INSERT INTO Podnik2
    SELECT Ev_cislo, Meno, ...
        FROM Podnik;
```

- **Príklad:** Doplňte tabuľku Fakulta stĺpcom s informačným číslom výskumnej úlohy na ktorej pracovník pracuje.

```
ALTER TABLE Fakulta
    ADD (Cislo_vu NUMBER(4));
UPDATE Fakulta
    SET Cislo_vu = 1023
        WHERE Funkcia = 'Docent'
                AND Katedra='KKUI';
UPDATE Fakulta
    SET Cislo_vu = 1025
        WHERE Funkcia = 'Asistent'
                OR Cislo_vu IS NULL;
```

➢ Zmena rozmeru stĺpca

ALTER TABLE <meno tabuľky>
MODIFY (<meno stĺpca> <typ > <nový rozmer>
[, <meno stĺpca> <typ > <nový rozmer>] …);

➢ **Príklad**: Zmeňte šírku stĺpca Mzda

ALTER TABLE Podnik
MODIFY Mzda NUMBER(8.2);

➢ Zrušenie celej tabuľky

DROP TABLE <meno tabuľky>;

➢ zruší sa aj definícia tabuľky