Databázové systémy

3. DDL príkazy, DML príkazy, WHERE klauzula, ORDER BY klauzula, vstavané funkcie, trojhodnotová logika

KKUI
Katedra kybernetiky
a umelej inteligencie

Opakovannie

- relačný model dát - záznamovo orientovaný model
- relačná databáza – údaje uložené v tabuľkách prepojených väzbami
- pojmy pre prácu s relačným modelom dát: relácia-tabuľka, n-tica-riadok, atribút-stĺpec, doména-množina všetkých hodnôt, identifikačný kľúč, árnosť(stupeň) relácie-počet atribútov objektu
- na prácu s DB slúži dátový jazyk SQL (DDL, DML, DCL, DTL)

Opakovanie

- **Dátové typy:**
  - znakové dátové typy : CHAR, VARCHAR2
  - číselné dátové typy : NUMBER
  - dátové typy pre kalendárne a časové hodnoty: DATE, TIMESTAMP
  - dátové typy pre objemné dáta: CLOB, BLOB

- **Integritné obmedzenia**
  - entitná integrita: PRIMARY KEY, UNIQUE
  - doménova integrita: NOT NULL, CHECK
  - referenčná integrita: FOREIGN KEY

Opakovanie

- Vytvorenie tabuľky:

```
CREATE TABLE <meno tabuľky>(
   <meno stĺpca1> <typ>(<dĺžka>),
   <meno stĺpca2> <typ>(<dĺžka>),...);

CREATE TABLE zamestnanec(
   id number(10) PRIMARY KEY,
   meno varchar2(20) NOT NULL,
   priezvisko varchar2(30),
   mesto varchar2(30),
   narodeny date,
   mzda number(5),
   veduci number(5),
   id_oddelenia number(3));
```

Tabuľka

- základný objekt databázy
- obsahuje konkrétne dáta
- je to dvojrozmerný objekt: dáta sú uložené v riadkoch a stĺpcoch
- je definovaná menom a stĺpcami, ktoré obsahuje
  - mená objektov databázy môžu obsahovať písmená, číslice a podčiarkovník, musia začínať písmenom, max. dĺžka je 30 znakov a ako identifikátory sa nesmú používať kľúčové slová (napr. *SELECT*)
  - meno objektu pozostáva z mena vlastníka a mena objektu: *student.predmety* pozn. ak sa odvolávame na svoje objekty, stačí používať iba na názov objektu
  - stĺpce tabuľky sú definované menom a množinou hodnôt, ktoré môžu byť doňho vložené; napr.: tabuľka *zamestnanec* ma stĺpec *priezvisko*, ktorý obsahuje reťazce s max. dĺžkou 30 znakov

Tabuľka

- nad objektami databázy vieme vykonávať tieto 3 typy príkazov: CREATE, ALTER, DROP t.j. vieme ich vytvoriť, zmeniť ich vlastnosti a zmazať

- SYNTAX pre vytvorenie tabuľky:

```
CREATE TABLE <meno tabuľky>
(<meno stĺpca1> <typ>(<dĺžka>),
 <meno stĺpca2> <typ>(<dĺžka>),
…);
```

Príklad:

```
CREATE TABLE zamestnanec(
                id number(10),
                meno varchar2(20),
                priezvisko varchar2(30),
                mesto varchar2(30),
                narodeny date,
                mzda number(5),
                veduci number(5),
                id_oddelenia number(3));
```

Práca s Tabuľkou

- pri tabuľke vieme meniť tieto vlastnosti:
  - pridanie stĺpca do tabuľky

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

  - odstránenie stĺpca

```
ALTER TABLE <meno tabuľky> DROP COLUMN <meno stĺpca>;
```

  - premenovanie stĺpca

```
ALTER TABLE <meno tabuľky> RENAME COLUMN <meno stĺpca> TO <nové meno stĺpca> ;
```

  - zmena dátoveho typu

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

  - pridanie obmedzenia nad stĺpcom tabuľky (bližšie info neskôr)

```
ALTER TABLE <meno tabuľky>  ADD/DROP CONSTRAINT <meno obmedzenia> ...;
```

  - premenovanie tabuľky

```
ALTER TABLE <meno tabuľky> RENAME TO <nové meno tabuľky>;
```

Práca s Tabuľkou

- odstránenie tabuľky

```
DROP <meno tabuľky> [CASCADE CONSTRAINTS];
```

- informácie (opis) o tabuľke predstavuje: zoznam atribútov (stĺpcov) tabuľky, ich údajové typy a ohraničenia (ak existujú).

```
DESC <meno tabuľky>;
DESCRIBE <meno tabuľky>;
```

SQL – vkladanie dát

```
INSERT INTO <meno tabuľky>
   VALUES (<zoznam hodnôt>);
INSERT INTO <meno tabuľky> (zoznam vybratých stĺpcov)
   VALUES (hodnoty pre vybratý zoznam);
```

- pridanie každého záznamu je cez INSERT
- jednotlivé hodnoty musia byť uvedené v poradí ako sú zadefinované atribúty tabuľky alebo v poradí ako uvedieme názvy stlpcov v príkaze INSER

```
INSERT INTO zamestnanec VALUES (12345, ‘Ján’,’Malý’
   ‘Košice’, ’11-mar-1978’, 700, 22, 5);
INSERT INTO zamestnanec (ev_cislo, meno, id_odd) VALUES
   (12378,’Pavel’,2);
```

SQL – aktualizácia dát

```
UPDATE <meno tabuľky>
SET <meno stĺpca1> = <výraz1>
[,<meno stĺpca2> = <výraz2> ...]
[WHERE  <podmienka>];
```

! Podmienka WHERE je nepovinná, ale jej nezadanie vedie k zmene všetkých hodnot v prisluhajúcom stĺpci

```
UPDATE zamestnanec
SET mzda = mzda+100
WHERE UPPER(mesto) <> 'KOSICE';
```

SQL – odstránenie dát

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

! nezadanie klauzuly WHERE vedie k zmazaniu všetkých dát v tabuľke

DELETE FROM zamestnanec WHERE priezvisko='Malý' AND meno='Ján';

Jazyk SQL – krátky súhrn

- každý objekt databázy môžme vytvoriť (CREATE), meniť jeho vlastnosti (ALTER) a zmazať ho (DROP)
- práca s dátami: vloženie nového údaju, aktualizácia údaju a zmazanie údaju
- UPOZORNENIE:
  - po každej zmene dát je potrebné túto zmenu potvrdiť príkazom COMMIT alebo sa vrátiť k pôvodnému stavu príkazom ROLLBACK
  - nepotvrdené zmeny vidíte iba vo svojej session, až keď ich potvrdíte stanú sa viditeľnými aj pre ostatných. Viac sa dozviete neskôr.

SQL – odstránenie dát

- odstráni všetky riadky z tabuľky

```
TRUNCATE TABLE <meno tabuľky>;
```

- TRUNCATE – DDL príkaz t.j. nie je transakčná operácia, dáta su hneď vymazané, nie je možné vykonať rollback

- rovnaký výsledok je možné dosiahnuť použitím

```
DELETE FROM <meno tabuľky>;
```

SQL – zobrazenie dát

- zobrazí údaje z tabuľky:

```
SELECT <zoznam stĺpcov v projekcii> / *
FROM <meno tabuľky>;
```

```
SELECT * FROM zamestnanec;
```

- zobrazí všetky stĺpce a všetky riadky z tabuľky zamestnanec

```
SELECT meno, priezvisko FROM zamestnanec;
```

- pre každý riadok tabuľky zamestnanec zobrazí iba údaje zo stĺpcov meno a priezvisko

SQL – zobrazenie dát

- pomocou príkazu DISTINCT vieme odstrániť duplikáty z výsledku t.j. každá hodnota z daného stĺpca sa zobrazí iba raz

```
SELECT DISTINCT mesto FROM zamestnanec;
```

  - výsledkom tohto príkazu bude zoznam miest, v ktorých žijú zamestnanci a každé mesto tam bude iba raz

- vymenovaním stĺpcov v SELECT príkaze vieme filtrovať stĺpce, ktoré sa majú zobraziť, pomocou klauzuly WHERE vieme zobraziť iba požadované riadky tabuľky

```
SELECT meno, priezvisko FROM zamestnanec WHERE
UPPER(mesto)=´KOŠICE´;
```

  - výsledkom bude zoznam mien zamestnancov, ktorý žijú v Košiciach

Poznámka:

Jazyk SQL je case-insensitive - jednotlivé klauzuly môžeme písať akýmikoľvek písmenami (SELECT=select=SelecT). To neplatí ak porovnávame hodnoty stĺpca t.j. Košice <> KOŠICE

SQL – ORDER BY

- triedenie záznamov podľa hodnôt niektorého atribútu resp. atribútov
  - ASC – vzostupne
  - DESC – zostupne
- pracuje aj s aliasmi

```
SELECT priezvisko, meno FROM zamestnanec ORDER BY
   priezvisko;
SELECT priezvisko, meno FROM zamestnanec ORDER BY 1, 2
   DESC;
```

SQL – základné operátory

- pri filtrovaní riadkov tabuľky na porovnavanie hodnôt používame operátory

- *relačné operátory:* = , >, <, <>, !=, <=, >=
- *aritmetické operátory:* +, -, * , /, MOD(a,b)
- || - spojenie dvoch reťazcov (konkatenácia)
- *IS [NOT] NULL* - test na hodnotu NULL
- *Logické operátory:* AND, OR, NOT

Príklad:

```
SELECT * FROM zamestnanec WHERE
  meno=´Ján´ AND priezvisko=´Malý´;
```

SQL – základné operátory

- ***BETWEEN***

```
SELECT * FROM zamestnanec WHERE mzda BETWEEN 50 AND 900;
```

- zobrazí tých zamestnancov, ktorých mzda je z uzavretého intervalu <50,900>

- ***IN***

```
SELECT * FROM zamestnanec WHERE mzda IN (500, 550, 600);
```

- zobrazí tých zamestnancov, ktorých mzda je 500, 550 alebo 600

- ***LIKE***

```
SELECT * FROM zamestnanec WHERE priezvisko LIKE ´%ová´;
```

- zobrazí tých zamestnancov, ktorých priezvisko končí na ‘ová’

Poznámka:

% - skupina ľubovoľných znakov dĺžky 0 až n (čiže v poslednom dopyte, by zobrazilo aj zamestnanca, ktorý má v stĺpci priezvisko zadanú hodnotu ová)

_ - jeden ľubovoľný znak

Vstavané funkcie

- pri zobrazovaní dát využívame aj množstvo už „predvytvorených“ – vstavaných funkcií

- Jednozáznamové funkcie (1 záznam -> 1 hodnota)
  - aritmetické funkcie
  - reťazcové funkcie
    - vracajúce reťazec
    - vracajúce numerické hodnoty
  - dátumové funkcie
  - funkcie pre konverziu dátových typov
- agregované funkcie (množina záznamov -> 1 hodnota)

Aritmetické funkcie

- numerický vstup -> numerická hodnota (NUMBER)

ABS(-15) ->15

CEIL(2,7) -> 3

FLOOR(2.8) -> 2

ROUND(10.589) -> 10.6

MOD(8, 3) -> 2

POWER(2, 3) -> 8
SQRT(25) -> 5
SIGN (-15) -> -1
GREATEST(105, 23, -85) -> 105
LEAST(105, 23, -85) -> -85

```
SELECT  meno, priezvisko, DECODE (predvoľba, 420, ´Česko´,
421, ´Slovensko´, 43, ´Rakúsko´) AS „Krajina“ FROM hovory;
```

- zobrazí zoznam mien (meno a priezvisko) a krajinu, kde telefonovali, pričom názov krajiny sa určí na základe predvoľby. Ak predvoľba bude iná ako vymenované, nahradí ju hodnotou NULL;

Reťazcové funkcie

- ***vracajúce reťazec***

UPPER('Abba') -> ABBA
LOWER('Abba') -> abba
SUBSTR('Abba', 2, 2) -> bb
TRIM(' abba ') -> abba
INITCAP(‘janko MRKVIČKA') -> Janko Mrkvička
REPLACE('Milan','lan','chal') -> Michal
CONCAT(´Anna´,´maria´) -> Annamaria

- ***vracajúce numerické hodnoty***

INSTR(´ABBA´,´B´, 2, 1) -> 2
LENGTH(´ABBA´) -> 4

Dátumové funkcie

- dátové typy: DATE a TIMESTAMP

SYSDATE - aktuálny dátum [a čas] dat. servra, formát určený parametrom NLS_DATE_FORMAT (v$nls_parameters)

CURRENT_DATE - aktuálny dátum v čas. pásme pripojenia klienta

CURRENT_DATE+1 - nasledujúci deň

CURRENT_TIMESTAMP - akt. časová značka

MONTHS_BETWEEN(date1, date2) – vráti počet mesiacov medzi dátumami date1 a date2

EXTRACT(month FROM CURRENT_DATE) - extrahuje hodnotu z položky typu DATETIME

NEXT_DAY('26-7-1996', 'NEDEĽA') - najbližší daný deň po uvedenom dátume

Funkcie pre konverziu dátových typov

KKUI
Katedra kybernetiky a umelej inteligencie

Z - DO	CHAR/ VARCHAR2	NUMBER	DATE
CHAR/ VARCHAR2	nepotrebné	TO_NUMBER	TO_DATE
NUMBER	TO_CHAR	nepotrebné	chybné
DATE	TO_CHAR	chybné	nepotrebné

SELECT TO_CHAR(narodeny, ´YYYY´) FROM zamestnanec;

SELECT TO_CHAR(SYSDATE, ´DD-MON-YYYY HH:MI:SS A.M.´) FROM dual;

SELECT TO_DATE(´Marec 11, 1978, 01:00´, ´Month dd, YYYY, HH24:MI´, ´NLS_DATE_LANGUAGE=´Slovak´) from dual;

SQL

- ***ALIAS*** : dočasné pomenovanie stĺpca alebo tabuľky

  napr: `SELECT ulica||`,`||mesto AS adresa FROM zamestnanec;`

  `SELECT z.ulica, z.mesto FROM zamestnanec z;`
- ***ROWNUM:*** poradové číslo výsledku daného riadku, hodnota sa priradí po vyfiltrovaní výsledku, ale pred agregovaním alebo triedením

  napr: `SELECT * FROM zamestnanec WHERE rownum < 11;`
- ***DUAL:*** jednoriadková a jednostĺpcová tabuľka použiteľná pri dopytoch, ktoré nepotrebujú klauzulu FROM napr: `SELECT 26564/24 FROM dual;`

Trojhodnotová logika

- NULL označuje prázdnu/nezadanú hodnotu
- agregované funkcie neberú NULL do úvahy
- pri práci používame IS [NOT] NULL

A	B	A AND B	A OR B	NOT A
TRUE	TRUE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	FALSE
TRUE	NULL	NULL	TRUE	FALSE
FALSE	TRUE	FALSE	TRUE	TRUE
FALSE	FALSE	FALSE	FALSE	TRUE
FALSE	NULL	FALSE	NULL	TRUE
NULL	TRUE	NULL	TRUE	NULL
NULL	FALSE	FALSE	NULL	NULL
NULL	NULL	NULL	NULL	NULL

SQL - SELECT

- vytvorenie tabuľky

```
CREATE TABLE <názov tabuľky> AS
SELECT <stĺpce> FROM <názov
tabuľky2> [WHERE <podmienky>];
```

- vloženie údajov z výsledku selectu

```
INSERT INTO <názov tabuľky>
SELECT <stĺpce> FROM <názov
tabuľky2>
[WHERE <podmienky>];
```

Odporučaná literatúra

- Database SQL Language Quick Reference
  https://docs.oracle.com/cd/E11882_01/server.112/e41085/toc.htm

- Database SQL Language Reference
  https://docs.oracle.com/cd/E11882_01/server.112/e41084/toc.htm

- Data Types
  https://docs.oracle.com/cd/E11882_01/server.112/e41084/sql_elements001.htm#SQLRF0021

- Constraint
  https://docs.oracle.com/cd/E11882_01/server.112/e41084/clauses002.htm#SQLRF52180

- ...