NÁVRH A PROGRAMOVÁNÍ DATABÁZÍ (14NDB)

rozsah: 0+2

zakončení: kl. zápočet

SYNONYMA

SYNONYMA I.

- alternativní název pro tabulky, pohledy, sekvence, procedury, funkce, package (balík) nebo i jiná synonyma
- synonymum pro tabulku:
  - INSERT, UPDATE, DELETE, SELECT
- synonymum lze vytvořit i pro neexistující objekt – v době vytvoření synonyma se nekontroluje existence objektu

SYNONYMA II.

CREATE [**OR REPLACE**] **SYNONYM**
název_synonyma

FOR [*schéma.*]*název_objektu*

```
CREATE SYNONYM zamestnanci FOR
  hr.employees;
```

```
SELECT * FROM zamestnanci;
```

SEKVENCE

SEKVENCE I.

= databázový "objekt" pro generování unikátních celých čísel

CREATE SEQUNCE *název_sekvence*

[**INCREMENT BY** *celé_číslo*]

[**START WITH** *celé_číslo*]

[**MINVALUE** *celé_číslo*]

[**MAXVALUE** *celé_číslo*]

[**CYCLE | NOCYCLE**] ;

SEKVENCE II.

```
CREATE SEQUENCE seq_pokus
  START WITH 1
  MINVALUE -250
  MAXVALUE 5
  CYCLE;
```

pozn.: není zcela zajištěno, že první vygenerované číslo bude mít hodnotu 1, další číslo bude 2, 3, 4 atd.

SEKVENCE III.

- číslo ze sekvence lze získat prostřednictvím pseudosloupců
  - **NEXTVAL** – vrací nově vygenerované číslo
  - **CURRVAL** – vrací aktuální (naposledy vygenerované) číslo sekvence pro dané spojení

SEKVENCE IV.

```
SELECT seq_pokus.NEXTVAL FROM DUAL;
```

```
SELECT seq_pokus.CURRVAL FROM DUAL;
```

```
INSERT INTO zamestnanci(id,prijmeni,pohlavi)
  VALUES(seq_pokus.NEXTVAL,'Náhlovský','M');
```

SEKVENCE V.

- změna definice sekvence

ALTER SEQUENCE *název_sekvence* ……;

- odstranění sekvence

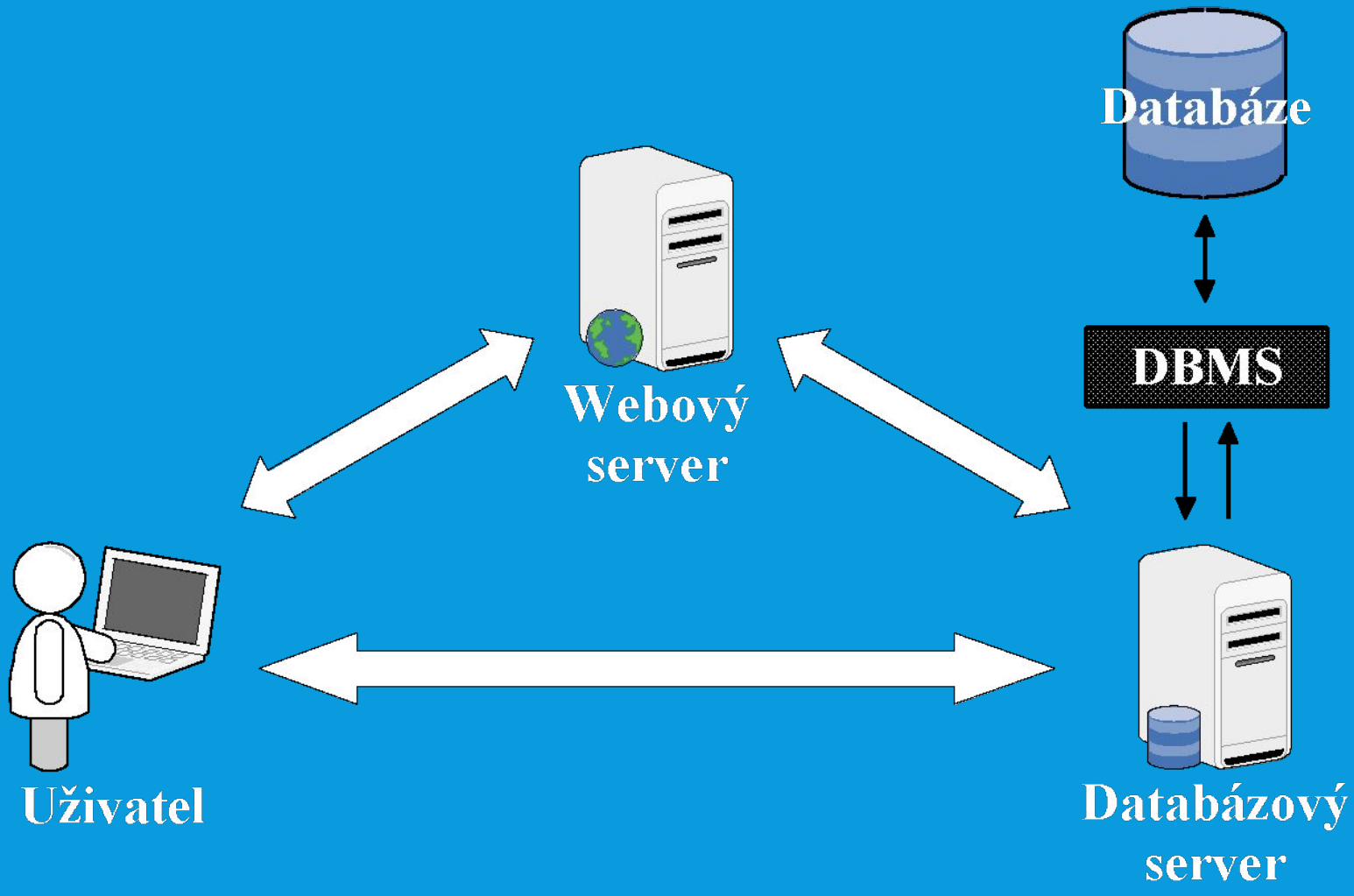
DROP SEQUENCE *název_sekvence*;

ÚKOL 1, 2

soubor 14NDB_cviceni_09.docx

PL/SQL

procedurální rozšíření jazyka SQL



ANONYMNÍ BLOK

[**DECLARE**

..... deklarační část]

BEGIN

..... výkonná část

[**EXCEPTION**

..... část pro zpracování výjimek]

END;

VÝJIMKY I.

```
BEGIN
....
EXCEPTION
   WHEN název_výjimky1 THEN příkazy1
   WHEN název_výjimky2 THEN příkazy2
     .
     .
   WHEN OTHERS THEN příkazyN+1
END;
```

VÝJIMKY II.

- předdefinované výjimky:
  - NO_DATA_FOUND – nebyla nalezena žádná data
  - TOO_MANY_ROWS – dotaz vrátil více než jeden záznam
  - ZERO_DIVIDE – dělení nulou

další viz dokumentace

```
DECLARE
 jmeno employees.first_name%TYPE;
 prijmeni employees.last_name%TYPE;
 id_zam employees.employee_id%TYPE := 50;
BEGIN
 SELECT last_name, first_name INTO prijmeni,
  jmeno
  FROM employees WHERE employee_id=id_zam;
EXCEPTION
  WHEN NO_DATA_FOUND THEN
     dbms_output.put_line(id_zam || ' je neplatné
  id');
END;
```

- pomocí funkcí **SQLCODE** a **SQLERRM** je možné získat číslo chyby a chybový text

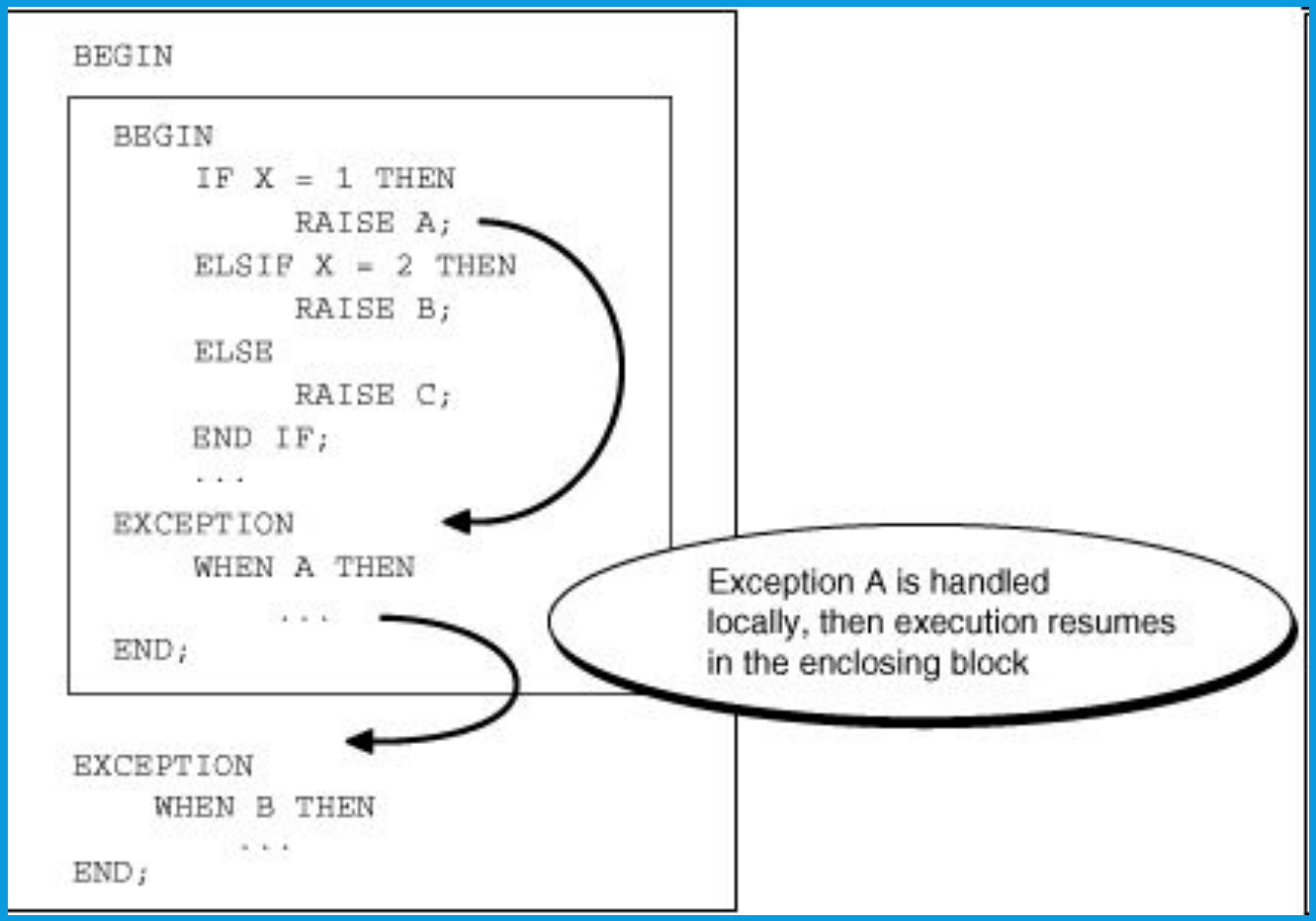
```
DECLARE
  name employees.last_name%TYPE;
  v_code NUMBER;
  v_errm VARCHAR2(64);
```

```
BEGIN
  SELECT last_name INTO name FROM employees
  WHERE employee_id = -1;
EXCEPTION
  WHEN OTHERS THEN
    v_code := SQLCODE;
    v_errm := SUBSTR(SQLERRM, 1 , 64);
    DBMS_OUTPUT.PUT_LINE('Error code ' || v_code || ': '
  || v_errm);
END;
```

VNOŘOVÁNÍ BLOKŮ

```
DECLARE
……
BEGIN
……
  BEGIN
     …
  EXCEPTION
     …
  END;
……
EXCEPTION
……
END;
```

ZACHYCENÍ VÝJIMKY



ÚKOL 3, 4

soubor 14NDB_cviceni_09.docx

KURZOR

- používá se při práci s více řádky vrácenými dotazem

1. deklarace kurzorů (v deklarační sekci)

   CURSOR *název_kurzoru* **IS** *dotaz_SELECT*;

2. otevření kurzoru (ve výkonné sekci)

   OPEN *název_kurzoru*;

3. výběr dat prostřednictvím kurzoru

   FETCH *název_kurzoru*
   INTO *seznam_proměnných* | *proměnná_typu_záznam*;

4. uzavření kurzoru

   CLOSE *název_kurzoru*;

```
DECLARE
    CURSOR kurzor_emp IS SELECT * FROM employees;
    record_emp kurzor_emp%ROWTYPE;
BEGIN
    OPEN kurzor_emp;
    FETCH kurzor_emp INTO record_emp;
    WHILE NOT kurzor_emp%NOTFOUND LOOP
      DBMS_OUTPUT.PUT_LINE('Číslo záznamu: ' ||
          kurzor_emp%ROWCOUNT);
      DBMS_OUTPUT.PUT_LINE('Jméno zaměstnance: ' ||
          record_emp.first_name);
      FETCH kurzor_emp INTO record_emp;
    END LOOP;
    CLOSE kurzor_emp;
END;
```

- informace o kurzoru je možné zjistit pomocí jeho atributů:

%FOUND - vrací TRUE, pokud kurzor ukazuje na nějaký záznam

%ISOPEN - vrací TRUE, pokud byl kurzor otevřen a nebyl uzavřen

%NOTFOUND - vrací TRUE, pokud kurzor neukazuje na žádný záznam

%ROWCOUNT - vrací pořadové číslo aktuálního záznamu, na kterém je kurzor umístěn

viz dokumentace

Práce s kurzorem při použití cyklu FOR

```
DECLARE
    CURSOR kurzor_emp IS SELECT * FROM employees;
    record_emp kurzor_emp%ROWTYPE;
BEGIN
    FOR record_emp IN kurzor_emp LOOP
      DBMS_OUTPUT.PUT_LINE('Číslo záznamu: ' ||
          kurzor_emp%ROWCOUNT);
      DBMS_OUTPUT.PUT_LINE('Jméno zaměstnance: ' ||
          record_emp.first_name);
    END LOOP;
END;
```

```
DECLARE
CURSOR kurzor_emp (emp_deptno
        employees.department_id%TYPE)
        IS SELECT * FROM employees
            WHERE department_id=emp_deptno;
record_emp kurzor_emp%ROWTYPE;
BEGIN
        FOR record_emp IN kurzor_emp(10) LOOP
            DBMS_OUTPUT.PUT_LINE('Číslo záznamu: ' ||
                kurzor_emp%ROWCOUNT);
            DBMS_OUTPUT.PUT_LINE('Jméno zaměstnance: ' ||
                 record_emp.first_name);
        END LOOP;
END;
```

```
DECLARE
CURSOR kurzor_emp (emp_deptno
        employees.department_id%TYPE)
        IS SELECT * FROM employees
            WHERE department_id=emp_deptno;
record_emp kurzor_emp%ROWTYPE;
BEGIN
        FOR record_emp IN kurzor_emp(&id_oddeleni) LOOP
            DBMS_OUTPUT.PUT_LINE('Číslo záznamu: ' ||
                kurzor_emp%ROWCOUNT);
            DBMS_OUTPUT.PUT_LINE('Jméno zaměstnance: ' ||
                 record_emp.first_name);
        END LOOP;
END;
```

ÚKOL 5, 6

soubor 14NDB_cviceni_09.docx