Cleverlance SQL

Manipulace s daty

Manipulace s daty

- Vkládání záznamů do tabulky
- Aktualizace záznamů v tabulce
- Odstranění záznamu z tabulky

INSERT

- Vkládání jednoho záznamu
- Syntaxe:

```
INSERT INTO   tabulka [(sloupec [, sloupec...])]
VALUES     (hodnota [, hodnota...]);
```

- Seznam sloupců můžeme vynechat pokud budeme vkládat hodnoty do všech sloupců, pořadí hodnot pak musí odpovídat pořadí sloupců v tabulce
- Znakové a datumové proměnné musí být ohraničeny pomocí apostrofů

Příklad

- Založení nového zaměstnance:
  - zamestnanec_id 123
  - jmeno a prijmeni Milan Hruby
  - pozice Cukrar
  - nadrizeny_id 26
  - oddeleni 6
  - zakladni plat 13000

```
INSERT INTO zamestnanci (zamestnanec_id, jmeno, prijmeni, pozice, nadrizeny_id, datum_nastupu, oddeleni, zakladni_plat) VALUES (123, 'Milan', 'Hruby', 'Cukrar', 26, '06-05-2005', 6, 13000);
```

Vkládání hodnoty NULL

Dvě metody, jak vložit prázdnou hodnotu:

- explicitní - použitím hodnoty NULL

```
INSERT INTO platy (id_zamestnance, mesic, rok, premie) VALUES (5, 12, 1982, NULL);
```

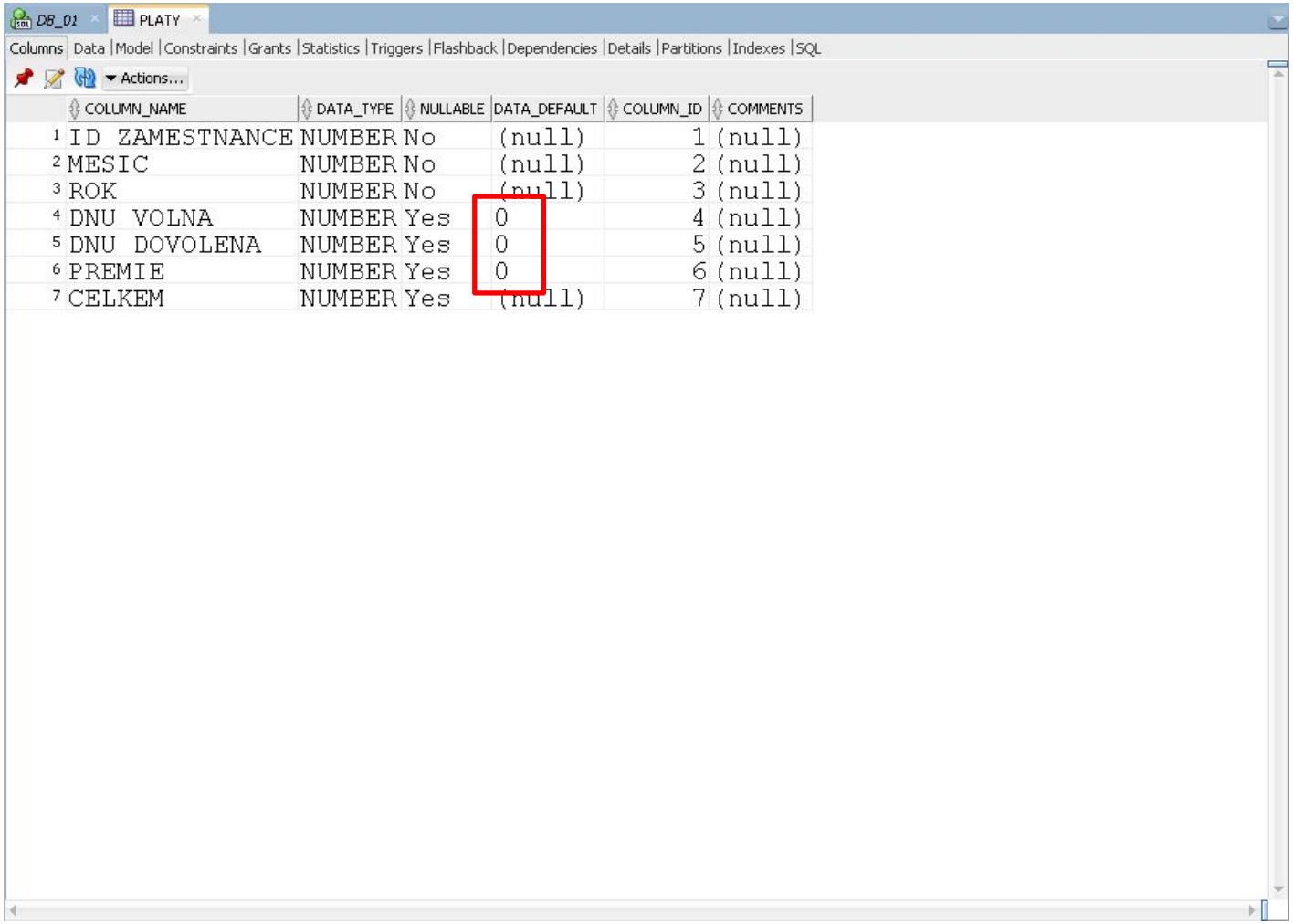
- implicitní - vynecháme sloupec v seznamu sloupců

```
INSERT INTO platy (id_zamestnance, mesic, rok) VALUES (5,12, 1982);
```

Defaultní hodnoty

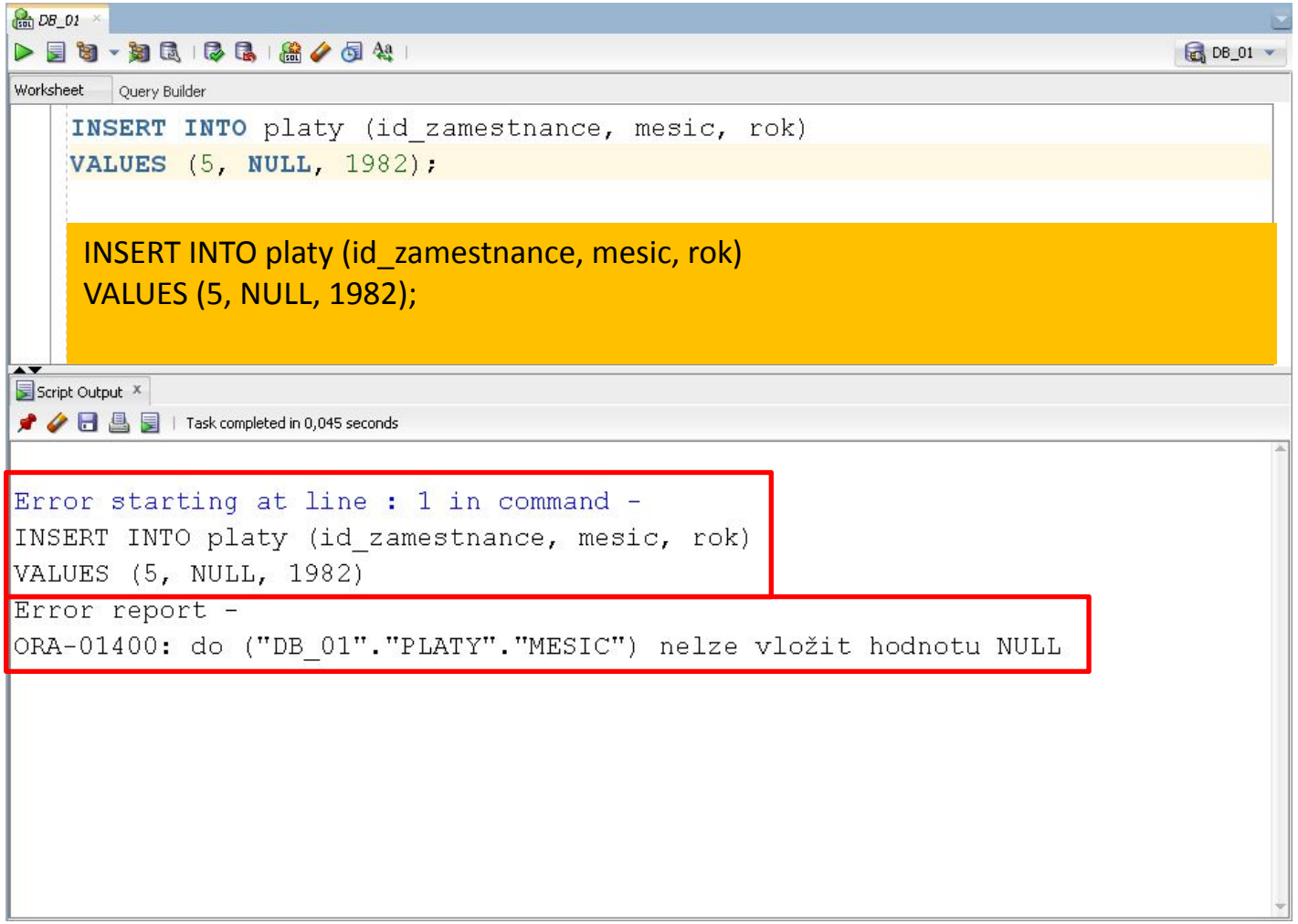
Pokud použijeme implicitní metodu pro vložení prázdné hodnoty na sloupec pro který je definována defaultní hodnota, pak se záznam uloží s definovanou defaultní hodnotou

Default



	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID_ZAMESTNANCE	NUMBER	No	(null)	1	(null)
2	MESIC	NUMBER	No	(null)	2	(null)
3	ROK	NUMBER	No	(null)	3	(null)
4	DNU_VOLNA	NUMBER	Yes	0	4	(null)
5	DNU_DOVOLENA	NUMBER	Yes	0	5	(null)
6	PREMIE	NUMBER	Yes	0	6	(null)
7	CELKEM	NUMBER	Yes	(null)	7	(null)

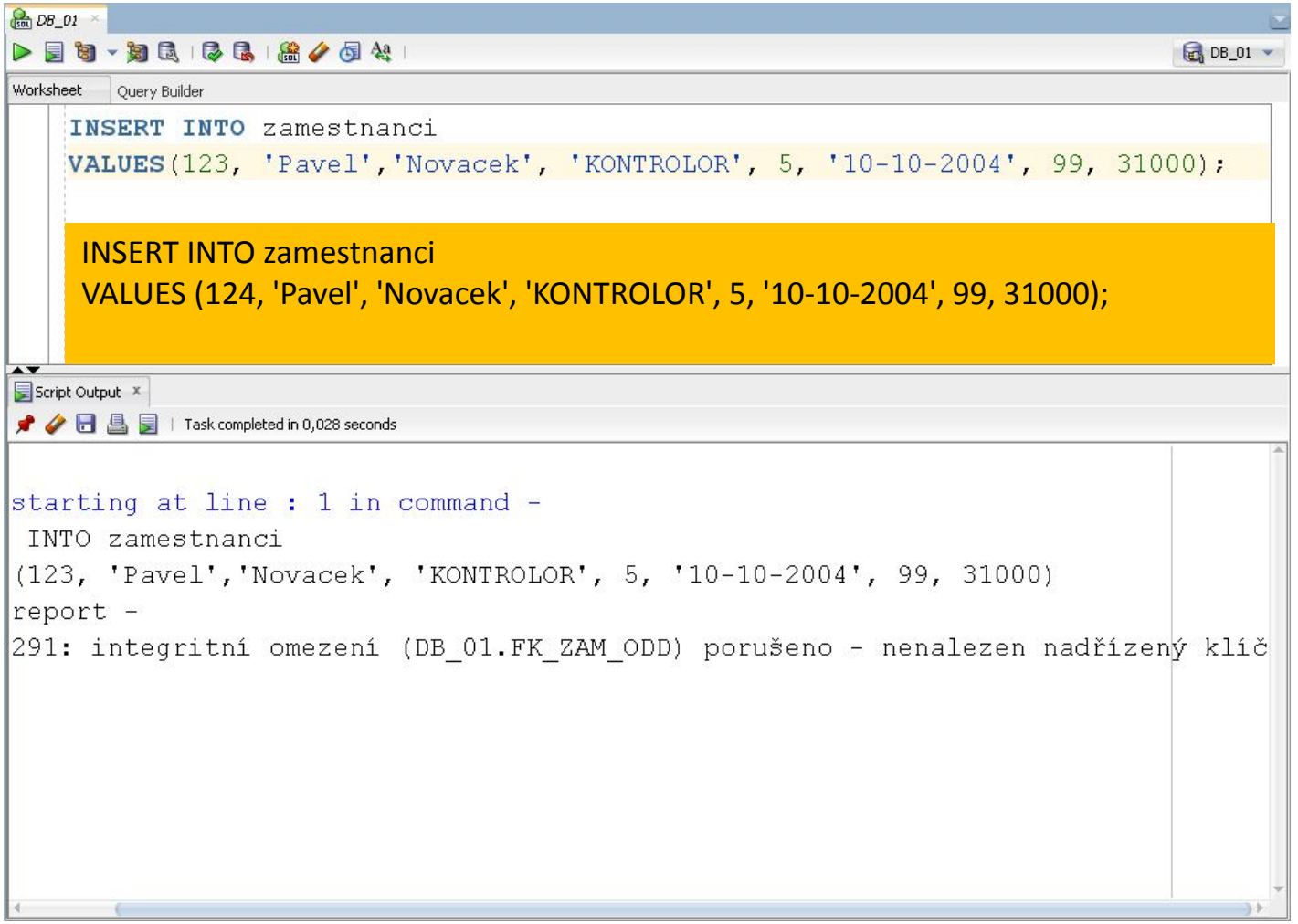
Omezení NOT NULL



FOREIGN KEY

- Cizí klíč – FOREIGN KEY
- Hodnota vkládaná do tohoto sloupce musí existovat v rodičovské tabulce
- Pokud sloupec není součástí primárního klíče je možné vložit hodnotu NULL (pokud na sloupci není omezení NOT NULL)

FOREIGN KEY


DB_01
Worksheet
Query Builder
INSERT INTO zamestnanci
VALUES(123, 'Pavel','Novacek', 'KONTROLOR', 5, '10-10-2004', 99, 31000);
INSERT INTO zamestnanci
VALUES (124, 'Pavel', 'Novacek', 'KONTROLOR', 5, '10-10-2004', 99, 31000);
Script Output
Task completed in 0,028 seconds
starting at line : 1 in command -
INTO zamestnanci
(123, 'Pavel','Novacek', 'KONTROLOR', 5, '10-10-2004', 99, 31000)
report -
291: integritní omezení (DB_01.FK_ZAM_ODD) porušeno - nenalezen nadřízený klíč

Příklad

Bylo založeno nové oddělení a zároveň byl přijat nový pracovník, který bude toto oddělení vést

Založte oba dva nové záznamy

– nové oddělení, které bude mít jako managera nového pracovníka

- nového pracovníka, který bude pracovat v novém oddělení

Automatické generování identifikátorů

- MS SQL
  - CREATE TABLE *tablename* (
    tablename_id INT **IDENTITY** PRIMARY KEY, ...)
- MySQL
  - CREATE TABLE *tablename* (
    tablename_id INTEGER **AUTO_INCREMENT** PRIMARY KEY, ...)
- Oracle
  - s využitím sekvence a triggerů

Sekvence (Oracle)

- Sekvence je numerický generátor hodnot

```
INSERT INTO oddeleni(oddeleni_id,
            oddeleni_nazev, lokalita_id)
VALUES      (odd_oddid_seq.NEXTVAL,
            'Support', 4);
1 row created.
```

- NEXTVAL vrací příští použitelné sekvenční číslo. Jde o unikátní hodnotu vždy když je voláno bez ohledu na uživatele
- CURRVAL obsahuje poslední sekvenční hodnotu vydanou pro aktuálního uživatele

Trigger (Oracle)

- Trigger na generovani primarnich klicu pro zamestnance?

- ```
  CREATE TABLE mytable (
    mytable_id INTEGER PRIMARY KEY,
    ... -- (other columns)
  );

  CREATE SEQUENCE mytable_seq;

  CREATE TRIGGER mytable_seq_trigger
  BEFORE INSERT ON mytable FOR EACH ROW
  BEGIN
    IF (:new.mytable_id IS NULL) THEN
      SELECT mytable_seq.nextval INTO :new.mytable_id
      FROM DUAL;
    END IF;
  END;
  /
  ```
  ```

UPDATE

- Umožňuje změnit stávající data
- Úprava sloupců vyjmenovaných v klauzuli SET se provede u všech záznamů, které odpovídají podmínkám v klauzuli WHERE
- Syntaxe:

```
UPDATE  tabulka
SET   sloupec=hodnota[,sloupec=hodnota,...]
[WHERE  podmínka];
```

Příklad UPDATE

- Zvyšte plat o 1000 Kč všem účetním.

```
UPDATE zamestnanci
SET zakladni_plat =
zakladni_plat + 1000
WHERE pozice='Ucetni';
```

```
UPDATE tabulka
SET sloupec=hodnota[,sloupec=hodnota,...]
[WHERE podmínka];
```

DELETE

- Příkaz DELETE slouží k odstranění záznamů z tabulky

```
DELETE [FROM]   tabulka
[WHERE     podmínka];
```

- Odstraní všechny záznamy, které splňují podmínky uvedené v klauzuli WHERE
- Lze smazat pouze celý řádek ne jen hodnoty jednotlivých sloupců

Příklad

- Smažeme zaměstnance se zaměstnaneckým číslem 123, kterého jsme založili v úvodu této lekce

```
DELETE FROM zamestnanci
WHERE zamestnanec_id=123;
```

Cvičení

1. Založte nový záznam v tabulce zaměstnanci, novému uživateli dejte své jméno a svůj datum nástupu, základní plat nevyplňujte
2. Aktualizujte nový záznam založený v předchozím kroku, změňte základní plat na 100 000.
3. Smažte zaměstnance, kterého jste v prvním kroku založili.