RBD LAB 6

Export danych

mysqldump sXXXX -p > DBdump_sXXXX.sql

Import danych

mysql -u sXXXX -p < DBdump_sXXXX.sql

Zmienne użytkownika

```
# Wyświetlić listę samochodów z data prod >= @x, gdzie x='2000-01-01'

SET @x='2000-01-01';
select * from samochod where data_prod>=@x;


# Wyświelić pole @x z tabeli samochod , gdzie @x ='id'

SET @x='id';
select id,@x from samochod;


#Zwrócenie nazwiska klienta o numerze pesel '80122412345' do zmiennej x

select nazwisko into @x from klient where pesel='80122412345';


#Przyklad zastosowania zmiennej do dynamicznej budowy zapytań

SET @x = "imie";
SET @s = CONCAT("SELECT ", @x, " FROM klient");
PREPARE stmt FROM @s;
EXECUTE stmt;
DEALLOCATE PREPARE stmt;
```

Procedury i funkcje

```
#TWORZENIE FUNKCJI HELLO
DROP FUNCTION IF EXISTS Hello;
DELIMITER ;;
CREATE FUNCTION Hello() RETURNS VARCHAR(20)
 BEGIN
   RETURN 'Hello world';
 END;
;;
DELIMITER ;
```

```
#WYWOLANIE HELLO()
mysql> select hello();


#TWORZENIE PROCEDURY HELLO

DROP PROCEDURE IF EXISTS Hello;

show warnings;

DELIMITER ;;
CREATE PROCEDURE Hello()
 BEGIN
   SELECT('Hello world');
 END;
;;
DELIMITER ;
```

#Funkcja bez parametru – dodawanie Kowalskiego jako Klienta kowalskiego

```
delimiter ;;

CREATE FUNCTION `dodaj_klienta`() RETURNS bool
BEGIN
insert into klient values ('','Wieslaw', 'Kowalski', '01010190098');
return true;

END;
;;
delimiter ;
```

#Funkcja z parametrem IN

```
delimiter ;;

CREATE FUNCTION `dodaj_klienta2`(imie varchar(20), nazwisko
varchar(40),pesel char(11)) RETURNS bool
BEGIN

insert into klient values ('',imie, nazwisko, pesel);
return true;

END;
;;
delimiter ;
```

wywołujemy:

```
select dodaj_klienta2('Andrzej','Biały','01234567890');
```

zwraca 1

 # Funkcja z parametrem IN + sprawdzenie klient istnieje?

```
IF search_condition THEN statement_list
    [ELSEIF search_condition THEN statement_list] ...
    [ELSE statement_list]
END IF
```

```
drop function if exists `dodaj_klienta3`;

CREATE FUNCTION `dodaj_klienta3`(imie varchar(20), nazwisko
varchar(40),pesel_ char(11)) RETURNS bool
BEGIN

if (select count(*) from klient where pesel=pesel_)>0 then
return false;
else
insert into klient values ('',imie, nazwisko, pesel_);
return true;
end if;

END;
```

#Procedura, która policzy samochody starsze i młodsze niż data_prod:

```
CREATE PROCEDURE `pokaz_samochody`(x date,out count_mlodsze int,out count_starsze
int)
BEGIN
select count(*) from samochod where data_prod>=x into count_mlodsze;
select count(*) from samochod where data_prod<x into count_starsze;
END;
```

wywołujemy:

```
call pokaz_samochody('2000-01-01',@a,@b);

select @a,@b;
```

#polecenie SHOW WARNINGS

#podgląd procedur i funkcji

```
select * from information_schema.routines where routine_type='PROCEDURE' or
routine_type='FUNCTION';

SHOW CREATE PROCEDURE nazwa;
SHOW CREATE FUNCTION nazwa;
SHOW PROCEDURE/FUNCTION STATUS;
```

TRIGERY

```
CREATE
    [DEFINER = { user | CURRENT_USER }]
    TRIGGER trigger_name
    trigger_time trigger_event
    ON tbl_name FOR EACH ROW
    trigger_body

trigger_time: { BEFORE | AFTER }

trigger_event: { INSERT | UPDATE | DELETE }
```

#dodac trigger, ktory bedzie notowal kazdy delete wiersza

```
delimiter ;;

CREATE TRIGGER t_klient_del AFTER DELETE ON klient
  FOR EACH ROW BEGIN
    INSERT INTO log values (now(),'usunieto wiersz z tabeli klient');
  END;
;;

delimiter ;
```

#podgląd triggera:

```
SELECT * FROM information_schema.TRIGGERS WHERE TRIGGER_SCHEMA = DATABASE()
AND TRIGGER_NAME = 't_klient_del';

SHOW TRIGGERS
```

#stworzyc trigger, ktory bedzie notowal w tabeli *log*, że wartość imienia klienta zmienila sie z x na y

```
delimiter ;;

DROP TRIGGER if exists t_klient_upd;

CREATE TRIGGER t_klient_upd AFTER UPDATE ON klient
  FOR EACH ROW BEGIN
    INSERT INTO log values (now(),concat('zmieniono wiersz w tabeli klient
imie z:',OLD.imie,' na :', NEW.imie));
  END;

;;

delimiter ;


update klient set imie='Andrzej' where id=9;

select * from log;
```

#trigger before insert – wykonujacy sprawdzenie i poprawianie wstawianej wartosci
stworzyć trigger, który będzie ustawiał wielkie litery na nazwisku każdego nowowstawianego klienta

```
DROP TRIGGER if exists t_klient_ins;

CREATE TRIGGER t_klient_ins BEFORE INSERT ON klient
  FOR EACH ROW BEGIN
SET NEW.imie=UPPER(NEW.imie);
  END;

insert into klient values ('','andrzej','iksinski','0101929201');

select * from klient;
```