Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

КАФЕДРА ИНФОРМАТИКИ

Лабораторная работа № 3

**«14. Система по заказу блюд в ресторане»**

Выполнил студент гр. 753504

Карницкий В. С.

Проверил

Алексеев И.Г.

**Минск, 2020**

* + **Таблица для backup**

CREATE TABLE IF NOT EXISTS `mydb`.`humanBackup` (

`idhumanBackup` INT NOT NULL AUTO\_INCREMENT,

`id` INT NOT NULL,

`Name2\_idName2` INT NOT NULL,

`Name1\_idName1` INT NOT NULL,

`Name3\_idName3` INT NOT NULL,

`login` VARCHAR(45) NULL,

`password` VARCHAR(45) NULL,

`status` ENUM('active', 'not active') NULL,

`role` ENUM('admin', 'user') NULL,

`active` ENUM('0', '1') NULL,

`blockedfrom` DATETIME NULL,

`blockedtill` DATETIME NULL,

PRIMARY KEY (`idhumanBackup`))

ENGINE = InnoDB;

* + **Триггер для создание backup для операции типа `update`**

create trigger mydb.beforeUpdateBackup

before update on mydb.human for each row

INSERT INTO mydb.humanbackup

(mydb.humanbackup.id, mydb.humanbackup.Name2\_idName2, mydb.humanbackup.Name1\_idName1, mydb.humanbackup.Name3\_idName3,

mydb.humanbackup.login, mydb.humanbackup.password, mydb.humanbackup.status, mydb.humanbackup.role,

mydb.humanbackup.active, mydb.humanbackup.blockedfrom, mydb.humanbackup.blockedtill)

SELECT \* FROM mydb.human WHERE mydb.human.id = old.id;

* + **Триггер для создание backup для операции типа `delete`**

create trigger mydb.beforeDeleteBackup

before delete on mydb.human for each row

INSERT INTO mydb.humanbackup

(mydb.humanbackup.id, mydb.humanbackup.Name2\_idName2, mydb.humanbackup.Name1\_idName1, mydb.humanbackup.Name3\_idName3,

mydb.humanbackup.login, mydb.humanbackup.password, mydb.humanbackup.status, mydb.humanbackup.role,

mydb.humanbackup.active, mydb.humanbackup.blockedfrom, mydb.humanbackup.blockedtill)

SELECT \* FROM mydb.human WHERE mydb.human.id = old.id;

* + **Отмена последней операции**

drop trigger if exists mydb.beforeDeleteBackup;

drop trigger if exists mydb.beforeUpdateBackup;

delete from mydb.human where mydb.human.id = @id;

insert into mydb.human

select mydb.humanbackup.id, mydb.humanbackup.Name2\_idName2, mydb.humanbackup.Name1\_idName1, mydb.humanbackup.Name3\_idName3, mydb.humanbackup.login, mydb.humanbackup.password, mydb.humanbackup.status, mydb.humanbackup.role, mydb.humanbackup.active, mydb.humanbackup.blockedfrom, mydb.humanbackup.blockedtill

from mydb.humanbackup where mydb.humanbackup.id = @id

and mydb.humanbackup.idhumanBackup =

(select max(mydb.humanbackup.idhumanBackup) from mydb.humanbackup);

delete from mydb.humanbackup where mydb.humanbackup.id = @id;

create trigger mydb.beforeUpdateBackup before update on mydb.human for each row INSERT INTO mydb.humanbackup (mydb.humanbackup.id, mydb.humanbackup.Name2\_idName2, mydb.humanbackup.Name1\_idName1, mydb.humanbackup.Name3\_idName3, mydb.humanbackup.login, mydb.humanbackup.password, mydb.humanbackup.status, mydb.humanbackup.role, mydb.humanbackup.active, mydb.humanbackup.blockedfrom, mydb.humanbackup.blockedtill)

SELECT \* FROM mydb.human WHERE mydb.human.id = old.id;

create trigger mydb.beforeDeleteBackup before delete on mydb.human for each row INSERT INTO mydb.humanbackup (mydb.humanbackup.id, mydb.humanbackup.Name2\_idName2, mydb.humanbackup.Name1\_idName1, mydb.humanbackup.Name3\_idName3, mydb.humanbackup.login, mydb.humanbackup.password, mydb.humanbackup.status, mydb.humanbackup.role, mydb.humanbackup.active, mydb.humanbackup.blockedfrom, mydb.humanbackup.blockedtill)

SELECT \* FROM mydb.human WHERE mydb.human.id = old.id;