-- проверка заполнения необходимых столбцов при вводе нового устройства

**DROP** **TRIGGER** **IF** **EXISTS** add\_devices;

**DELIMITER //**

**CREATE** **TRIGGER** add\_devices **BEFORE** **INSERT** **ON** inventory.deviсes

**FOR** **EACH** **ROW**

**BEGIN**

**IF**(**ISNULL**(**NEW**.device\_name) **AND**

**ISNULL**(**NEW**.description) **AND**

**ISNULL**(**NEW**.location\_id) **AND**

**ISNULL**(**NEW**.device\_type\_id) **AND**

**ISNULL**(**NEW**.device\_model\_id) **AND**

**ISNULL**(**NEW**.status\_id)) **THEN**

**SIGNAL** **SQLSTATE** '45000' **SET** MESSAGE\_TEXT = 'Отмена операции, необходимо заполнить поля: device\_name, description, location\_id, device\_type\_id, device\_model\_id, status\_id';

**END** **IF**;

**END** //

**DELIMITER ;**

-- создание архивной таблицы

**CREATE** **TABLE** logs (

table\_name **VARCHAR**(60) **NOT** **NULL**,

id **BIGINT**(50) **NOT** **NULL**,

name\_value **VARCHAR**(60) **NOT** **NULL**,

created\_time **DATETIME** **NOT** **NULL**

) ENGINE = ARCHIVE;

-- триггер каналы

**DROP** **TRIGGER** **IF** **EXISTS** channels;

**DELIMITER //**

**CREATE** **TRIGGER** channels\_log **AFTER** **INSERT** **ON** channels

**FOR** **EACH** **ROW**

**BEGIN**

**INSERT** **INTO** logs (table\_name, id, name\_value, created\_time)

**VALUES** ('channels', **NEW**.id, **NEW**.name, **NOW**());

**END** //

**DELIMITER ;**

-- триггер устройства

**DROP** **TRIGGER** **IF** **EXISTS** deviсes;

**DELIMITER //**

**CREATE** **TRIGGER** deviсes\_log **AFTER** **INSERT** **ON** deviсes

**FOR** **EACH** **ROW**

**BEGIN**

**INSERT** **INTO** logs (table\_name, id, name\_value, created\_time)

**VALUES** ('deviсes', **NEW**.id, **NEW**.device\_name, **NOW**());

**END** //

**DELIMITER ;**

-- триггер пользователи

**DROP** **TRIGGER** **IF** **EXISTS** users;

**DELIMITER //**

**CREATE** **TRIGGER** users\_log **AFTER** **INSERT** **ON** users

**FOR** **EACH** **ROW**

**BEGIN**

**INSERT** **INTO** logs (table\_name, id, first\_name, last\_name, created\_time)

**VALUES** ('users', **NEW**.id, **NEW**.first\_name, **NEW**.last\_name, **NOW**());

**END** //

**DELIMITER ;**

-- по имени канала возвращает провайдера, локации канала

**DROP** **PROCEDURE** **IF** **EXISTS** getChannelInfo;

**DELIMITER //**

**CREATE** **PROCEDURE** getChannelInfo(**IN** chname **varchar**(100))

**LANGUAGE** **SQL**

**NOT** **DETERMINISTIC**

**CONTAINS** **SQL**

**SQL** **SECURITY** **DEFINER**

COMMENT 'по имени канала возвращает провайдера, локации канала '

**BEGIN**

**SELECT** **DISTINCT** channels.name **AS** channels, providers.name **AS** providers, to\_loc.location\_name,from\_loc.location\_name

**FROM** channels

**JOIN** providers

**ON** channels.provider\_id = providers.id

**JOIN** location to\_loc

**ON** to\_loc.id = channels.to\_location\_id

**JOIN** location from\_loc

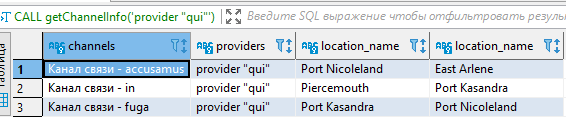
**ON** from\_loc.id = channels.from\_location\_id

**WHERE** providers.name **COLLATE** utf8mb4\_general\_ci = chname;

**END**//

**DELIMITER ;**

**CALL** getChannelInfo('provider "qui"');



-- по имени оборудования возвращает локацию сервис статус

**DROP** **PROCEDURE** **IF** **EXISTS** getDeviceInfo;

**DELIMITER //**

**CREATE** **PROCEDURE** getDeviceInfo(**IN** devname **varchar**(100))

**LANGUAGE** **SQL**

**NOT** **DETERMINISTIC**

**CONTAINS** **SQL**

**SQL** **SECURITY** **DEFINER**

COMMENT 'по имени оборудования возвращает локацию, сервис, статус'

**BEGIN**

**SELECT** d.device\_name, l.location\_name, s.name **AS** service, st.name **AS** status

**FROM** deviсes d, location l, services s, status st **WHERE** d.device\_name **COLLATE** utf8mb4\_general\_ci = devname **AND** l.id = d.location\_id **AND** s.id = d.services\_id **AND** st.id = d.status\_id;

**END**//

**DELIMITER ;**

**CALL** getDeviceInfo('Totamenimu-055');

