CREATE TABLE account\_history (

id serial,

edit\_time timestamp DEFAULT now(),

tabel\_name VARCHAR(255),

operation VARCHAR(255),

users VARCHAR(255) DEFAULT current\_user,

new\_val json,

old\_val json

);

CREATE FUNCTION change\_trigger() RETURNS trigger AS $$

BEGIN

IF TG\_OP = 'INSERT'

THEN

INSERT INTO account\_history (tabel\_name, operation, new\_val)

VALUES (TG\_RELNAME, TG\_OP, row\_to\_json(NEW));

RETURN NEW;

ELSIF TG\_OP = 'UPDATE'

THEN

INSERT INTO account\_history (tabel\_name, operation, new\_val, old\_val)

VALUES (TG\_RELNAME, TG\_OP,

row\_to\_json(NEW), row\_to\_json(OLD));

RETURN NEW;

ELSIF TG\_OP = 'DELETE'

THEN

INSERT INTO account\_history (tabel\_name, operation, old\_val)

VALUES (TG\_RELNAME, TG\_OP, row\_to\_json(OLD));

RETURN OLD;

END IF;

END;

$$ LANGUAGE 'plpgsql' SECURITY DEFINER;

CREATE FUNCTION change\_trigger() RETURNS trigger AS $$

BEGIN

IF TG\_OP = 'INSERT'

THEN

INSERT INTO account\_history (tabel\_name,user\_id ,operation, new\_val,account\_number,account\_id)

VALUES (TG\_RELNAME, NEW."user\_id" ,TG\_OP, NEW."balance",NEW."account\_number",NEW."id");

RETURN NEW;

ELSIF TG\_OP = 'UPDATE'

THEN

INSERT INTO account\_history (tabel\_name, user\_id ,operation, new\_val, old\_val,account\_number,account\_id)

VALUES (TG\_RELNAME, OLD."user\_id" ,TG\_OP,NEW."balance", OLD."balance",NEW."account\_number",NEW."id");

RETURN NEW;

ELSIF TG\_OP = 'DELETE'

THEN

INSERT INTO account\_history (tabel\_name,user\_id , operation, old\_val,account\_number,account\_id)

VALUES (TG\_RELNAME, OLD."user\_id" ,TG\_OP, OLD."balance",OLD."account\_number",OLD."id");

RETURN OLD;

END IF;

END;

$$ LANGUAGE 'plpgsql' SECURITY DEFINER;

CREATE TRIGGER history BEFORE INSERT OR UPDATE OR DELETE ON account

FOR EACH ROW EXECUTE PROCEDURE change\_trigger();