МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

КАФЕДРА СИСТЕМОТЕХНІКИ

Звіт

З практичної роботи №3

На тему: «Реалізація програмної логіки й забезпечення цілісності зв’язків за допомогою тригерів для високонавантаженої баз даних на платформі СУБД MySQL»

з дисципліни «Проектування високонавантажених систем зберігання даних»

Виконав:

ст. гр. ІТКНу-19-2

Марковець Н.С.

Перевірив викладач:

Коваленко А.І.

Харків 2020

Мета: Набуття практичних навичок з розробки тригерів для підтримки цілісності зв’язків, модифікації даних і забезпечення основних бізнес-процесів високонавантаженої інформаційної системи. Формування необхідних практичних умінь для аналізу плану виконання SQL-запитів за допомогою оператора EXPLAIN. Формування необхідних практичних умінь для створення тригерів, з урахуванням особливостей реалізації логіки роботи інтерфейсу високонавантаженої інформаційної системи зберігання даних.

Тема індивідуального завдання: Інформаційна система «Надання послуг вантажоперевезень»

Таблиця 1 – Перелік доступних операцій тригерів

Доступність операцій тригера	BEFORE INSERT	AFTER INSERT	BEFORE UPDATE	AFTER UPDATE	BEFORE DELETE	AFTER DELETE
1. NEW доступний – Так/Ні	так	так	так	так	ні	ні
2. OLD доступний – Так/Ні	ні	ні	так	так	так	так
3. Доступ до зміни полів NEW – Так/Ні						
4. Значення автоінкремента (PK) доступно – Так/Ні	ні	так	ні	так	ні	Ні
5. Доступ до значень полів та їх значень (DEFAULT), що явно в інструкціях INSERT, UPDATE, DELETE не фігурували – Так/Ні	ні	так	ні	так	ні	так
6. Можливість скасування операцій INSERT / / UPDATE / DELETE – Так/Ні	так	так	так	так	так	так

Таблиця 2 – Переваги використання тригерів

№	Аналізовані параметри	Тригер	SQl-запит
1	Принцип виконання SQL-коду в СУБД MySQL	Збережений код	
2	Вплив на апаратні ресурси	Впливає на апаратні ресурси сервера	Впливає на все
3	Вплив на мережевий трафік	менший	більший
4	Можливість, що-небудь автоматизувати	можна	можна
5	SQL-синтаксис (обмеження)	За допомогою ключового слова WHEN	
6	Принцип (фізичне місце) зберігання SQL-коду	Сервер	Клієнтська частина

Скріншот схеми фізичної моделі бази даних з таблицями типу InnoDB у нотації IDEF1X представлено на рисунку 1.

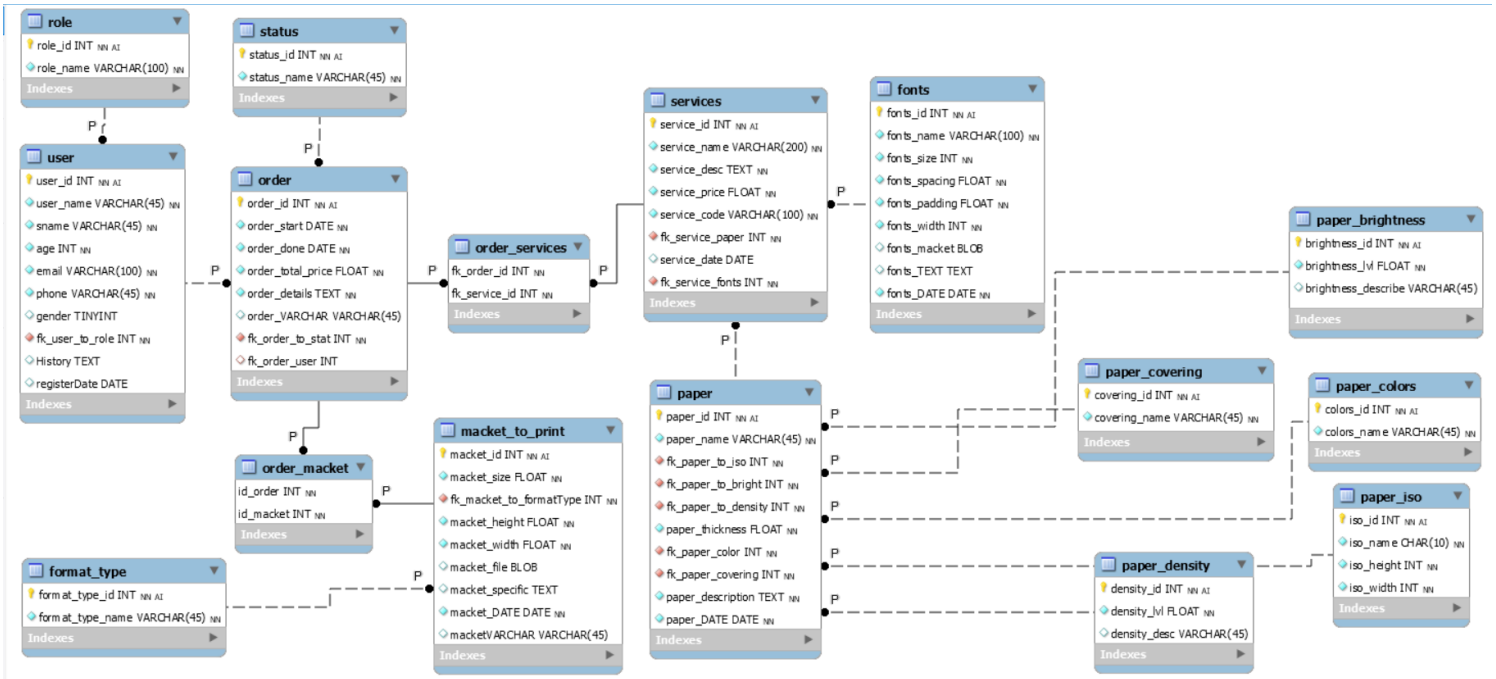

Рисунок 1 – Фізична модель даних

Завдання 3.1 Таблиця з переліком типів посилальної цілісності зв'язків за зовнішнім ключем для всіх таблиць типу MyIsam, реалізованих за допомогою тригерів.

Таблиця 3 – Перелік типів посилальної цілісності за допомогою тригерів

№	Ім'я таблиці 1, зовнішній ключ	Ім'я таблиці 2, первинний ключ	Тип посилальної цілісності для таблиці 1	Тип посилальної цілісності для таблиці 2	Тригер
1	User, fk_user_to_role	Role, role_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
2	User, fk_user_to_role	Role, role_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
3	User, fk_user_to_role	Role, role_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete
4	Order, Fk_order_to_stat	Status, status_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
5	Order, Fk_order_to_stat	Status, status_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
6	Order, Fk_order_to_stat	Status, status_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete
7	Order, Fk_order_to_user	User, user_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
8	Order, Fk_order_to_ user	User, user_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
9	Order, Fk_order_to_ user	User, user_id	ON DELETE RESTRICT	ON DELETE SET NULL	Тригер Before delete
10	Order_macket, Id_order	Order, order_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
11	Order_macket, Id_order	Order, order_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
12	Order_macket, Id_order	Order, order_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete
13	Order_macket, Id_order	Macket_to_print, macket_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
14	Order_macket, Id_order	Macket_to_print, macket_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
15	Order_macket, Id_order	Macket_to_print, macket_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete
16	Macket_to_print, fk_macket_to_formatType	Format_type, format_type_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
17	Macket_to_print, fk_macket_to_formatType	Format_type, format_type_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
18	Macket_to_print, fk_macket_to_formatType	Format_type, format_type_id	ON DELETE RESTRICT	ON DELETE SET NULL	Тригер Before delete

Продовження таблиці 3

19	Paper, fk_paper_to_iso	Paper_iso, iso_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
20	Paper, fk_paper_to_iso	Paper_iso, iso_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
21	Paper, fk_paper_to_iso	Paper_iso, iso_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete
22	Paper, fk_paper_to_bright	Paper_brightness, brightness_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
23	Paper, fk_paper_to_bright	Paper_brightness, brightness_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
24	Paper, fk_paper_to_bright	Paper_brightness, brightness_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete
25	Paper, fk_paper_to_density	Paper_density, density_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
26	Paper, fk_paper_to_density	Paper_density, density_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
27	Paper, fk_paper_to_density	Paper_density, density_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete
28	Paper, fk_paper_color	Paper_color, color_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
29	Paper, fk_paper_color	Paper_color, color_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
30	Paper, fk_paper_color	Paper_color, color_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete
31	Paper, fk_paper_coverign	Paper_covering, covering_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
32	Paper, fk_paper_coverign	Paper_covering, covering_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
33	Paper, fk_paper_coverign	Paper_covering, covering_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete
34	Services, fk_service_paper	Paper, paper_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
35	Services, fk_service_paper	Paper, paper_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
36	Services, fk_service_paper	Paper, paper_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete
37	Services, fk_service_fonts	Font, font_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
38	Services, fk_service_fonts	Font, font_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
39	Services, fk_service_fonts	Font, font_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete
40	Order_services, fk_order_id	Order, order_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
41	Order_services, fk_order_id	Order, order_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
42	Order_services, fk_order_id	Order, order_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete

Продовження таблиці 3

43	Order_services, fk_service_id	Services, service_id	ON INSERT RESTRICT	ON INSERT RESTRICT	Тригер Before insert
44	Order_services, fk_service_id	Services, service_id	ON UPDATE RESTRICT	ON UPDATE RESTRICT	Тригер Before update
45	Order_services, fk_service_id	Services, service_id	ON DELETE RESTRICT	ON DELETE RESTRICT	Тригер Before delete

1. Тригер для таблиці Role типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `role_BEFORE_UPDATE` BEFORE UPDATE ON `role` FOR EACH ROW BEGIN
    if exists(select 1  from my.`user` u where fk_user_to_role = old.role_id) and old.role_id != new.role_id
    then
        signal sqlstate '45000' set message_text = 'can not update data ';
    end if;
END
```

Рисунок 1. Код тригера

Результат спрацювання тригера :

```
update my.`role`
set role_id = 2
where role_id = 1
```

Output

Action Output

#	Time	Action	Message
1	13:45:48	update my.`role` set role_id = 2 where role_id = 1	Error Code: 1644. can not update data

Рисунок 2. Результат роботи тригера

2. Тригер для таблиці Role типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `role_BEFORE_DELETE` BEFORE DELETE ON `role` FOR EACH ROW BEGIN
    if exists(select 1  from my.`user` where fk_user_to_role = old.role_id)
    then
        signal sqlstate '45000' set message_text = 'can not delete ';
    end if;
END
```

Рисунок 3. Код тригера

Результат спрацювання тригера:

```
delete from my.`role`
where role_id = 2;
```

```
Output
Action Output
#  Time      Action                                  Message
1  13:51:22  delete from my.`role` where role_id = 2  Error Code: 1644. can not delete
```

Рисунок 4. Результат роботи тригера

3. Тригер для таблиці User типу **Before Insert**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `user_BEFORE_INSERT` BEFORE INSERT ON `user` FOR EACH ROW BEGIN
        if not exists (select 1 from my.`role` where role_id = new.fk_user_to_role)
    then
        signal sqlstate '45000' set message_text = 'can not insert row. There is no the role_id';
    end if;
END
```

Рисунок 5. Код тригера

```
INSERT INTO `my`.`user`
(`user_id`, `user_name`, `sname`, `age`, `email`, `phone`, `gender`, `fk_user_to_role`, `History`, `registerDate`)
VALUES ('4001', 'Annya', 'Melnick', '20', 'test@', 'test', '1', '3', 'test', '2012-12-12');

Output
Action Output
#  Time      Action                                                                                                  Message
1  14:03:41  INSERT INTO 'my'.'user' ('user_id', 'user_name', 'sname', 'age', 'email', 'phone', 'gender', 'fk_user_to...  Error Code: 1644. can not insert row. There is no the role_id
```

Рисунок 6. Результат роботи тригера

4. Тригер для таблиці User типу **Before update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `user_BEFORE_UPDATE` BEFORE UPDATE ON `user` FOR EACH ROW BEGIN
        if not exists (select 1 from my.`role` where role_id = new.fk_user_to_role)
    then
        signal sqlstate '45000' set message_text = 'can not update row. There is no the role_id';
    end if;
END
```

Рисунок 7. Код тригера

Рисунок 8. Результат роботи тригера

5. Тригер для таблиці User типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `user_BEFORE_DELETE` BEFORE DELETE ON `user` FOR EACH ROW BEGI
    if exists (select 1 from my.`order` o where o.fk_order_user = old.user_id)
    then
        update my.`order` o
        set o.fk_order_user = null
        where o.fk_order_user = old.user_id;
    end if;
END
```

Рисунок 9. Код тригера

Результат роботи представлено на рисунках 10, 11.

```
delete from my.user
where user_id = 4001;

select * from my.user
where user_id between 3999 and 4003;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

user_id	user_name	sname	age	email	phone	gender	fk_user_to_role	History	registerDate
3999	John	Holt	20	JohnHolt1gmail.com	(+380)-74-243-9136	0	2	The user made sales for price:234NE^MIJON^U	2001-02-11
4000	Charles	Henderson	49	CharlesHenderson4gmail.com	(+380)-57-958-3738	1	2	The user made sales for price:571JDXQZZGZPD	2018-12-17
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 10. Видалення користувача

order_id	order_start	order_done	order_total_price	order_details	order_VARCHAR	fk_order_to_stat	fk_order_user
51	2012-12-12	2012-12-13	4444	test	test	2	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 11. Зміна ключа для таблиці замовлень на null

Згідно з законодавчими нормами користувач має право до керування своїми персональними даними.

6. Тригер для таблиці Status типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `status_BEFORE_UPDATE` BEFORE UPDATE ON `status` FOR EACH ROW BEGIN
    if exists(select 1  from my.`order` where fk_order_to_stat = old.status_id) and old.status_id != new.status_
    then
        signal sqlstate '45000' set message_text = 'can not update data ';
    end if;
END
```

Рисунок 12. Код тригера

```
update my.`status`
set status_id = 5
where status_id = 1;

select * from my.status;
```

```
Output
Action Output
#  Time      Action                                                  Message
1  15:23:51  update my.`status` set status_id = 5 where status_id = 1  Error Code: 1644. can not update data
```

Рисунок 13. Результат роботи тригера

7. Тригер для таблиці Status типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `status_BEFORE_DELETE` BEFORE DELETE ON `status` FOR EACH ROW BEGIN
    if exists(select 1  from my.`order` o where o.fk_order_to_stat = old.status_id)
    then
        signal sqlstate '45000' set message_text = 'can not delete data. There are fk to this status ';
    end if;
END
```

Рисунок 14. Код тригера

```
delete from my.status
where status_id = 1;

select * from my.status;
```

```
Output
Action Output
#  Time      Action                                  Message
1  15:25:46  delete from my.status where status_id = 1  Error Code: 1644. can not delete data. There are fk to this status
```

Рисунок 15. Результат роботи тригера

8. Тригер для Order типу **Before insert**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `order_macket_BEFORE_INSERT` BEFORE INSERT ON `order_macket` FOR EACH ROW BEGIN
if not exists
(select 1 from my.`order` o where o.order_id = new.id_order) or
 not exists (select 1 from my.macket_to_print m where m.macket_id = new.id_macket)
    then
        signal sqlstate '45000' set message_text = 'Can not insert new row';
    end if;
END
```

Рисунок 16. Код тригера

```
INSERT INTO `my`.`order`
(`order_id`, `order_start`, `order_done`, `order_total_price`, `order_details`, `order_VARCHAR`, `fk_order_to_stat`, `fk_order_user`)
 VALUES ('52', '2012-12-12', '2012-12-12', '4444', 'test ', 'test ', '5', '4');
```

Output

Action Output

#	Time	Action	Message
1	16:28:38	INSERT INTO 'my'.'order' ('order_id', 'order_start', 'order_done', 'order_total_price', 'order_details', 'ord...	Error Code: 1644. Can not insert. There are no user or status

Рисунок 17. Результат роботи тригера

9. Тригер для Order типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `order_macket_BEFORE_UPDATE` BEFORE UPDATE ON `order_macket` FOR EACH ROW BEGIN
if not exists
(select 1 from my.`order` o where o.order_id = new.id_order) or
not exists (select 1 from my.macket_to_print m where m.macket_id = new.id_macket)
    then
        signal sqlstate '45000' set message_text = 'Can not update row ';
    end if;
END
```

Рисунок 18. Код тригера

```
update my.order
set fk_order_to_stat = 4005
where order_id = 51;

select * from my.order where order_id = 51;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

order_id	order_start	order_done	order_total_price	order_details	order_VARCHAR	fk_order_to_stat	fk_order_user
51	2012-12-12	2012-12-12	4444	test	test	4	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

order 27 | order 34

Output

Action Output

#	Time	Action	Message
1	16:34:25	update my.order set fk_order_to_stat = 4005 where order_id = 51	Error Code: 1644. Can not update data
2	16:34:38	select * from my.order where order_id = 51	1 row(s) returned

Рисунок 19. Результат роботи тригера

10. Тригер для Order типу **Before delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `order_macket_BEFORE_DELETE` BEFORE DELETE ON `order_macket` FOR EACH ROW BEGIN
if exists
(select 1 from my.`order` o where o.order_id = old.id_order) or
not exists(select 1 from my.macket_to_print m where m.macket_id = old.id_macket)
    then
        signal sqlstate '45000' set message_text = 'Нельзя удалить запись ';
    end if;
END
```

Рисунок 20. Код тригера

```
DELETE FROM `my`.`order` WHERE (`order_id` = '51');
```

Output

Action Output

#	Time	Action	Message
1	16:31:17	DELETE FROM `my`.`order` WHERE (`order_id` = '51')	Error Code: 1644. Can not delete order

Рисунок 21. Результат роботи

11. Тригер для Order_Macket типу **Before insert**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `order_macket_BEFORE_INSERT` BEFORE INSERT ON `order_macket` FOR EACH ROW BEGIN
if not exists
(select 1 from my.`order` o where o.order_id = new.id_order) or
 not exists (select 1 from my.macket_to_print m where m.macket_id = new.id_macket)
    then
        signal sqlstate '45000' set message_text = 'Can not insert new row';
    end if;
END
```

Рисунок 22. Код тригера

```
insert into my.order_macket (id_order, id_macket)
values (2, 61);
```

Output

Action Output

#	Time	Action	Message
1	17:49:25	insert into my.order_macket (id_order, id_macket) values (2, 61)	Error Code: 1644. Can not insert new row

Рисунок 23. Результат роботи

12. Тригер для Order_Macket типу **Before Update**

Код тригера представлено на рисунку 24

```
CREATE DEFINER=`root`@`localhost` TRIGGER `order_macket_BEFORE_UPDATE` BEFORE UPDATE ON `order_macket` FOR EACH ROW BEGIN
if not exists
(select 1 from my.`order` o where o.order_id = new.id_order) or
not exists (select 1 from my.macket_to_print m where m.macket_id = new.id_macket)
    then
        signal sqlstate '45000' set message_text = 'Can not update row ';
    end if;
END
```

Рисунок 24. Код тригера

```
update my.order_macket
set id_macket = 61
where id_order = 2 and id_macket = 10;
select * from my.order_macket;
```

Output

Action Output

#	Time	Action	Message
1	17:50:45	update my.order_macket set id_macket = 61 where id_order = 2 and id_macket = 10	Error Code: 1644. Can not update row

Рисунок 25. Результат роботи

13. Тригер для Order_Macket типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `order_macket_BEFORE_DELETE` BEFORE DELETE ON `order_macket` FOR EACH ROW BEGIN
if exists
(select 1 from my.`order` o where o.order_id = old.id_order) or
not exists(select 1 from my.macket_to_print m where m.macket_id = old.id_macket)
    then
        signal sqlstate '45000' set message_text = 'Нельзя удалить запись ';
    end if;
END
```

Рисунок 26. Код тригера

```
DELETE FROM `my`.`order_macket` WHERE (`id_order` = '2') and (`id_macket` = '10');
```

Output

Action Output

#	Time	Action	Message
1	17:48:39	DELETE FROM 'my'.'order_macket' WHERE ('id_order' = '2') and ('id_macket' = '10')	Error Code: 1644. Нельзя удалить запись

Рисунок 27. Код тригера

14. Тригер для Macket_to_print типу **Before Insert**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `macket_to_print_BEFORE_INSERT` BEFORE INSERT ON `macket_to_print` FOR EACH ROW BEGIN
if not exists
( select 1 from my.format_type f where f.format_type_id = new.fk_macket_to_formatType)
and new.fk_macket_to_formatType is not null
    then
        signal sqlstate '45000' set message_text = 'Нельзя вставить запись. Отсутствие формата ';
    end if;
END
```

Рисунок 28. Код тригера

```
INSERT INTO `my`.`macket_to_print`
(`macket_id`, `macket_size`, `fk_macket_to_formatType`, `macket_height`, `macket_width`, `macket_specific`, `macket_DATE`, `macketVARCHAR`)
 VALUES ('51', '23', '23', '234', '22', 'test', '2010-12-12', 'test');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

macket_id	macket_size	fk_macket_to_formatType	macket_height	macket_width	macket_file	macket_specific	macket_DATE	macketVARCHAR
31	13.1	4	764.8	760.3	NULL	Ut non enim eleifend felis pretium feugiat. Viva…	2007-04-22	uncommonly considered
32	44.9	2	370.3	1434.7	NULL	In hac habitasse platea dictumst. Curabitur at l…	2004-03-04	set especially prosperous
33	34.1	1	507.9	1193.9	NULL	Curabitur a felis in nunc fringilla tristique. Morbi …	2009-09-13	Sons at park
34	27.9	2	633.5	2437.5	NULL	Phasellus gravida semper nisi. Nullam vel sem. P…	2014-07-20	mr meet as fact like.
35	28.6	3	645.9	2463.1	NULL	Sed hendrerit. Morbi ac felis. Nunc egestas, au…	2015-10-19	Unpacked now declared
36	3.5	4	1087.5	3356.2	NULL	Donec interdum, metus et hendrerit aliquet, dol…	2001-04-27	put you confined
37	26.1	1	1348.8	3738.9	NULL	Morbi nec metus. Phasellus blandit leo ut odio. …	2018-06-05	daughter improved
38	10.8	3	812.9	3136.6	NULL	Quisque libero metus, condimentum nec, tempo…	2004-08-16	Celebrated imprudence
39	49.1	2	1738.4	2523.3	NULL	Fusce fermentum. Nullam cursus lacinia erat. Pr…	2015-11-18	few interested
40	49.1	4	1225.1	2313.3	NULL	Ut tincidunt tincidunt erat. Etiam feugiat lorem n…	2001-02-07	especially reasonable
41	45.3	5	1087.2	3957.5	NULL	Praesent egestas neque eu enim. In hac habita…	2009-03-04	Wonder bed elinor
42	42.7	4	1706.9	935.4	NULL	Sed augue ipsum, egestas nec, vestibulum et, …	2000-07-12	It want gave west
43	46.3	3	1998.9	3028.2	NULL	Proin faucibus arcu quis ante. In consectetuer t…	2004-07-16	into high no in.
44	37.8	1	1312.1	965.6	NULL	Nulla neque dolor, sagittis eget, iaculis quis, mol…	2015-02-08	Bringing so sociable
45	8.6	1	922.7	599.2	NULL	Suspendisse non nisl sit amet velit hendrerit rutr…	2012-04-10	felicity supplied mr

ket_to_print 1 ×

Output

Action Output

#	Time	Action	Message
1	18:03:12	INSERT INTO `my`.`macket_to_print` (`macket_id`, `macket_size`, `fk_macket_to_formatType`, `macket…	Error Code: 1644. Нельзя вставить запись. Отсутствие формата

Рисунок 29. Результат роботи

15. Тригер для Macket_to_print типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `macket_to_print_BEFORE_UPDATE` BEFORE UPDATE ON `macket_to_print` FOR EACH ROW BEGIN
if not exists
( select 1 from my.format_type f where f.format_type_id = new.fk_macket_to_formatType)
    then
        signal sqlstate '45000' set message_text = 'Нельзя обновить запись. Отсутствие формата ';
    end if;
END
```

Рисунок 30. Код тригера

```
1  update my.macket_to_print
2  set fk_macket_to_formatType = 20
3  where macket_id = 51;
4  select * from my.macket_to_print;
5
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

macket_id	macket_size	fk_macket_to_formatType	macket_height	macket_width	macket_file	macket_specific	macket_DATE	macketVARCHAR
38	10.8	3	812.9	3136.6	NULL	Quisque libero metus, condimentum nec, tempo…	2004-08-16	Celebrated imprudence
39	49.1	2	1738.4	2523.3	NULL	Fusce fermentum. Nullam cursus lacinia erat. Pr…	2015-11-18	few interested
40	49.1	4	1225.1	2313.3	NULL	Ut tincidunt tincidunt erat. Etiam feugiat lorem n…	2001-02-07	especially reasonable
41	45.3	5	1087.2	3957.5	NULL	Praesent egestas neque eu enim. In hac habita…	2009-03-04	Wonder bed elinor
42	42.7	4	1706.9	935.4	NULL	Sed augue ipsum, egestas nec, vestibulum et, …	2000-07-12	It want gave west
43	46.3	3	1998.9	3028.2	NULL	Proin faucibus arcu quis ante. In consectetuer t…	2004-07-16	into high no in.
44	37.8	1	1312.1	965.6	NULL	Nulla neque dolor, sagittis eget, iaculis quis, mol…	2015-02-08	Bringing so sociable
45	8.6	1	922.7	599.2	NULL	Suspendisse non nisl sit amet velit hendrerit rutr…	2012-04-10	felicity supplied mr
46	34.2	1	789.2	3767.1	NULL	Curabitur suscipit suscipit tellus. Praesent vesti…	2009-06-10	September suspicion
47	11.7	4	833.9	3450.4	NULL	Maecenas egestas arcu quis ligula mattis placer…	2005-03-21	far him two acuteness pe…
48	3.7	2	1661.2	2125.4	NULL	Aliquam erat volutpat. Etiam vitae tortor. Morbi …	2010-08-12	Covered as an examine
49	3.7	4	1598.3	1140.2	NULL	Etiam ut purus mattis mauris sodales aliquam. C…	2014-05-18	Ye astonished friendship
50	37.8	1	1081.2	2686.9	NULL	Sed mollis, eros et ultrices tempus, mauris ipsum…	2008-08-22	remarkably no
51	23	23	234	22	NULL	test	2010-12-12	test
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

macket_to_print 1

Output

Action Output

#	Time	Action	Message
1	18:05:54	update my.macket_to_print set fk_macket_to_formatType = 20 where macket_id = 51	Error Code: 1644. Нельзя обновить запись. Отсутствие формата

Рисунок 31. Результат роботи тригера

16. Тригер для Macket_to_print типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `macket_to_print_BEFORE_DELETE` BEFORE DELETE ON `macket_to_print` FOR EACH ROW BEGIN
if exists
(select 1 from my.order_macket om where om.id_macket = old.macket_id)
    then
        signal sqlstate '45000' set message_text = 'Нельзя удалить запись ';
    end if;
END
```

Рисунок 32. Код тригера

```
1  delete from my.macket_to_print
2  where macket_id = 10;
3  select * from my.macket_to_print;
4
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

macket_id	macket_size	fk_macket_to_formatType	macket_height	macket_width	macket_file	macket_specific	macket_DATE	macketVARCHAR
38	10.8	3	812.9	3136.6	NULL	Quisque libero metus, condimentum nec, tempo…	2004-08-16	Celebrated imprudence
39	49.1	2	1738.4	2523.3	NULL	Fusce fermentum. Nullam cursus lacinia erat. Pr…	2015-11-18	few interested
40	49.1	4	1225.1	2313.3	NULL	Ut tincidunt tincidunt erat. Etiam feugiat lorem n…	2001-02-07	especially reasonable
41	45.3	5	1087.2	3957.5	NULL	Praesent egestas neque eu enim. In hac habita…	2009-03-04	Wonder bed elinor
42	42.7	4	1706.9	935.4	NULL	Sed augue ipsum, egestas nec, vestibulum et, …	2000-07-12	It want gave west
43	46.3	3	1998.9	3028.2	NULL	Proin faucibus arcu quis ante. In consectetuer t…	2004-07-16	into high no in.
44	37.8	1	1312.1	965.6	NULL	Nulla neque dolor, sagittis eget, iaculis quis, mol…	2015-02-08	Bringing so sociable
45	8.6	1	922.7	599.2	NULL	Suspendisse non nisl sit amet velit hendrerit rutr…	2012-04-10	felicity supplied mr
46	34.2	1	789.2	3767.1	NULL	Curabitur suscipit suscipit tellus. Praesent vesti…	2009-06-10	September suspicion
47	11.7	4	833.9	3450.4	NULL	Maecenas egestas arcu quis ligula mattis placer…	2005-03-21	far him two acuteness pe…
48	3.7	2	1661.2	2125.4	NULL	Aliquam erat volutpat. Etiam vitae tortor. Morbi …	2010-08-12	Covered as an examine
49	3.7	4	1598.3	1140.2	NULL	Etiam ut purus mattis mauris sodales aliquam. C…	2014-05-18	Ye astonished friendship

macket_to_print 1

Output

Action Output

#	Time	Action	Message
1	18:07:54	delete from my.macket_to_print where macket_id = 10	Error Code: 1644. Нельзя удалить запись

Рисунок 33. Результат роботи тригера

17. Тригер для Format_type типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `format_type_BEFORE_UPDATE` BEFORE UPDATE ON `format_type` FOR EACH ROW BEGIN
    if exists(select 1  from my.macket_to_print m where m.fk_macket_to_formatType = old.format_type_id)
    then
        signal sqlstate '45000' set message_text = 'can not update data ';
    end if;
END
```

Рисунок 34. Код тригера

```
update my.format_type
set format_type_id = 6
where format_type_id = 5;
select * from my.format_type;
```

Output

Action Output

#	Time	Action	Message
1	18:38:48	update my.format_type set format_type_id = 6 where format_type_id = 5	Error Code: 1644. can not update data

Рисунок 35. Результат роботи тригера

18. Тригер для Format_type типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `format_type_BEFORE_DELETE` BEFORE DELETE ON `format_type` FOR EACH ROW BEGIN
    if exists(select 1  from my.macket_to_print where fk_macket_to_formatType = old.format_type_id)
    then
        update my.macket_to_print m
        set fk_macket_to_formatType = null
        where fk_macket_to_formatType = old.format_type_id;
    end if;
END
```

Рисунок 36. Код тригера

format_type_id	format_type_name
1	.GIF
2	.PNG
3	.JPG
4	.JPEG
5	.BMP
6	.BitMap
NULL	NULL

Рисунок 37. Значення батьківської таблиці до видалення

macket_id	macket_size	fk_macket_to_formatType	macket_height	macket_width	macket_file	macket_specific	macket_DATE	macketVARCHAR
51	20	6	209	200	NULL	test	2020-10-30	test
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 38. Відповідний запис в дочірній таблиці

```
1 •  delete from my.format_type where format_type_id = 6;
2 •  select * from my.format_type;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

format_type_id	format_type_name
1	.GIF
2	.PNG
3	.JPG
4	.JPEG
5	.BMP
NULL	NULL

format_type 14

Output

Action Output

#	Time	Action	Message
1	20:35:44	delete from my.format_type where format_type_id = 6	1 row(s) affected
2	20:35:44	select * from my.format_type	5 row(s) returned

Рисунок 39. Процес видалення запису батьківської таблиці

macket_id	macket_size	fk_macket_to_formatType	macket_height	macket_width	macket_file	macket_specific	macket_DATE	macketVARCHAR
51	20	NULL	209	200	NULL	test	2020-10-30	test
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 40. Оновлені дані дочірньої таблиці

19. Тригер для Order_services типу **Before Insert**

```
1 •  CREATE DEFINER=`root`@`localhost` TRIGGER `order_services_BEFORE_INSERT` BEFORE INSERT ON `order_services` FOR EACH ROW BEGIN
2    if not exists
3    (select 1 from my.`order` o where o.order_id = new.fk_order_id) or not
4    exists (select 1 from my.`services` where service_id = new.fk_service_id)
5        then
6            signal sqlstate '45000' set message_text = 'Can not insert new row';
7        end if;
8    END
```

Рисунок 41. Код тригера

```
1 •  INSERT INTO `my`.`order_services` (`fk_order_id`, `fk_service_id`) VALUES ('61', '60');
```

Output

Action Output

#	Time	Action	Message
1	22:58:26	INSERT INTO `my`.`order_services` (`fk_order_id`, `fk_service_id`) VALUES ('61', '60')	Error Code: 1644. Can not insert new row

Рисунок 42. Результат роботи тригера

20. Тригер для Order_services типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `order_services_BEFORE_UPDATE` BEFORE UPDATE ON `order_services` FOR EACH ROW BEGIN
if exists
(select 1 from my.`order` o where o.order_id = new.fk_order_id) or
exists (select 1 from my.`services` where service_id = new.fk_service_id)
    then
        signal sqlstate '45000' set message_text = 'Can not update';
    end if;
END
```

Рисунок 43. Код тригера

```
update my.order_services
set fk_order_id = 30
where fk_order_id = 47;

select * from order_services;
```

```
Output
Action Output
#  Time      Action                                                          Message
1  22:56:44  update my.order_services set fk_order_id = 30 where fk_order_id = 47  Error Code: 1644. Can not update
```

Рисунок 44. Результат роботи тригера

21. Тригер для Order_services типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `order_services_BEFORE_DELETE` BEFORE DELETE ON `order_services` FOR EACH ROW BEGIN
if exists
(select 1 from my.`order` o where o.order_id = old.fk_order_id) or
exists (select 1 from my.`services` where service_id = old.fk_service_id)
    then
        signal sqlstate '45000' set message_text = 'Can not delete data';
    end if;
END
```

Рисунок 45. Код тригера

```
delete from my.order_services
where fk_order_id = 47
```

```
Output
Action Output
#  Time      Action                                          Message
1  22:59:39  delete from my.order_services where fk_order_id = 47  Error Code: 1644. Can not delete data
```

Рисунок 46. Результат роботи тригера

22. Тригер для Services типу **Before Insert**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `services_BEFORE_INSERT` BEFORE INSERT ON `services` FOR EACH ROW BEGIN
if not exists
(select 1 from my.fonts f where f.fonts_id = new.fk_service_fonts) or
not exists (select 1 from my.paper where paper.paper_id = new = fk_service_paper)
    then
        signal sqlstate '45000' set message_text = 'Нельзя вставить запись ';
    end if;
END
```

Рисунок 47. Код тригера

```
1   INSERT INTO `my`.`services`
2   (`service_id`, `service_name`, `service_desc`, `service_price`, `service_code`, `fk_service_paper`, `service_date`, `fk_service_fonts`)
3   VALUES ('52', 'TST', 'TEST', '100', '#3000000', '30', '2020-11-11', '60');
4
```

```
Output
Action Output
#  Time      Action                                                                                  Message
1  21:59:42  INSERT INTO `my`.`services` (`service_id`, `service_name`, `service_desc`, `service_price`, `service_co...  Error Code: 1644. Нельзя вставить запись
```

Рисунок 48. Код тригера

23. Тригер для Services типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `services_BEFORE_DELETE` BEFORE DELETE ON `services` FOR EACH ROW BEGIN
if  exists
(select 1 from my.order_services os where os.fk_service_id = old.service_id) or
exists (select 1 from my.paper where paper.paper_id = old.fk_service_paper)
    then
        signal sqlstate '45000' set message_text = 'Can not delete this service ';
    end if;
END
```

Рисунок 49. Код тригера

```
1   delete from my.services where service_id = 49
```

```
Output
Action Output
#  Time      Action                                  Message
1  22:01:12  delete from my.services where service_id = 49  Error Code: 1644. Can not delete this service
```

Рисунок 50. Результат роботи тригера

24. Тригер для Services типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `services_BEFORE_UPDATE` BEFORE UPDATE ON `services` FOR EACH ROW BEGIN
if not exists
(select 1 from my.fonts f where f.fonts_id = new.fk_service_fonts) or
not exists (select 1 from my.paper where paper.paper_id = new = fk_service_paper)
    then
        signal sqlstate '45000' set message_text = 'Нельзя вставить запись ';
    end if;
END
```

Рисунок 51. Код тригера

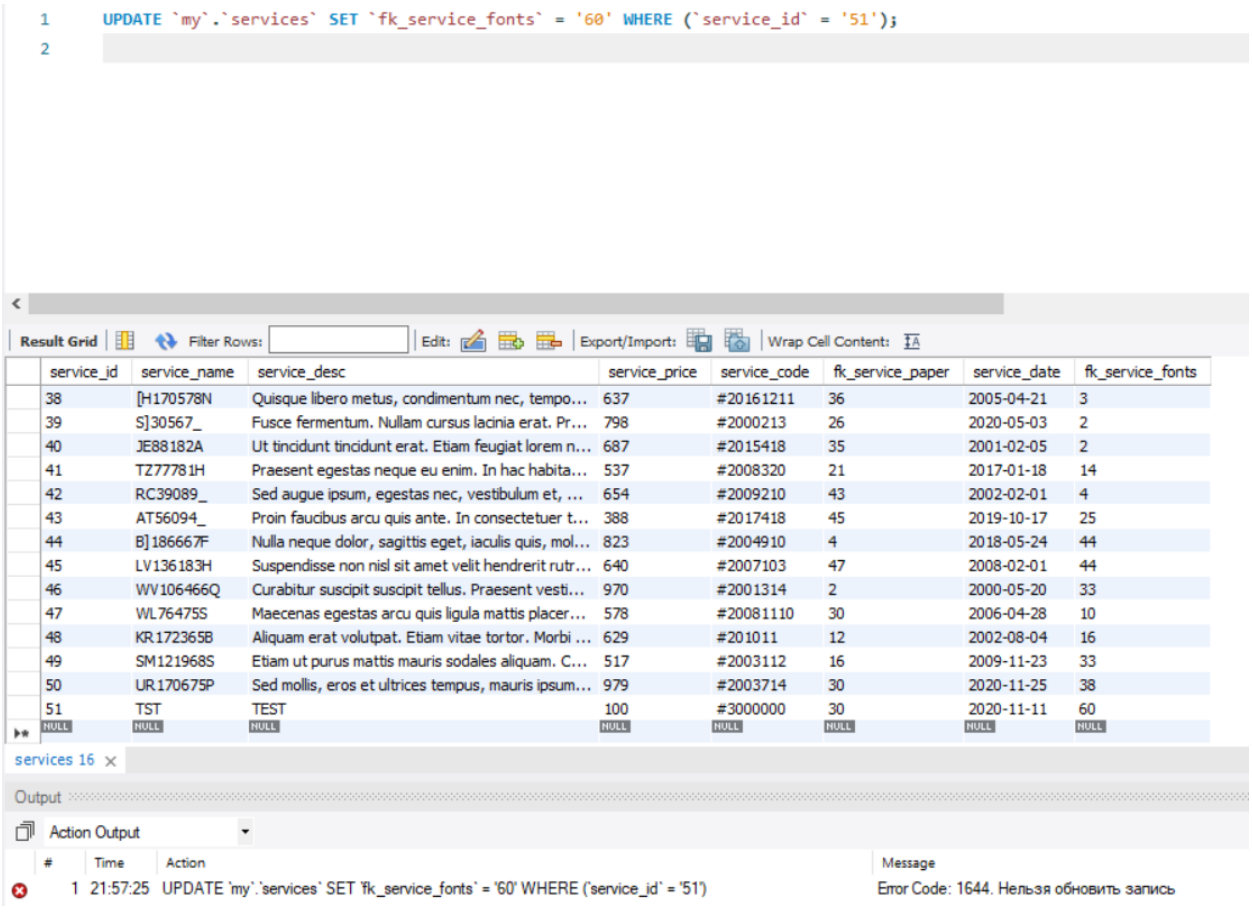

Рисунок 52. Результат роботи тригера

25. Тригер для Fonts типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `fonts_BEFORE_UPDATE` BEFORE UPDATE ON `fonts` FOR EACH ROW BEGIN
    if exists(select 1  from my.`services` s where s.fk_service_fonts = old.fonts_id) and old.fonts_id != new.fonts_id
    then
        signal sqlstate '45000' set message_text = 'can not update data ';
    end if;
END
```

Рисунок 53. Код тригера

```
update my.fonts
set fonts_id = 50
where fonts_id = 8;

select * from fonts;
```

```
Output
Action Output
#  Time      Action                                                  Message
1  22:30:58  update my.fonts set fonts_id = 50 where fonts_id = 8    Error Code: 1644. can not update data
```

Рисунок 54. Результат роботи тригера

26. Тригер для Fonts типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `fonts_BEFORE_DELETE` BEFORE DELETE ON `fonts` FOR EACH ROW BEGIN
if exists(select 1  from my.`services` where fk_service_fonts = old.fonts_id)
    then
        signal sqlstate '45000' set message_text = 'can not delete ';
    end if;
END
```

Рисунок 55. Код тригера

```
delete from my.fonts
where fonts_id = 8;

select * from fonts;
```

```
Output
Action Output
#  Time      Action                                  Message
1  22:31:42  delete from my.fonts where fonts_id = 8  Error Code: 1644. can not delete
```

Рисунок 56. Результат роботи тригера

27. Тригер для Paper типу **Before Insert**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `paper_BEFORE_INSERT` BEFORE INSERT ON `paper` FOR EACH ROW BEGIN
if
not exists (select 1 from my.paper_brightness where brightness_id = new.fk_paper_to_bright) or
not exists (select 1 from my.paper_covering where covering_id = new.fk_paper_covering) or
not exists (select 1 from my.paper_iso where iso_id = new.fk_paper_to_iso) or
not exists (select 1 from my.paper_density where density_id = new.fk_paper_to_density) or
not exists (select 1 from my.paper_colors  where colors_id = new.fk_paper_color)
    then
        signal sqlstate '45000' set message_text = 'Нельзя вставить запись ';
    end if;
END
```

Рисунок 57. Код тригера

Рисунок 58. Результат роботи тригера

28. Тригер для Paper типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `paper_BEFORE_UPDATE` BEFORE UPDATE ON `paper` FOR EACH ROW BEGIN
if
exists (select 1 from my.paper_brightness where brightness_id = new.fk_paper_to_bright) or
exists(select 1 from my.paper_covering where covering_id = new.fk_paper_covering) or
exists(select 1 from my.paper_iso where iso_id = new.fk_paper_to_iso) or
exists(select 1 from my.paper_density where density_id = new.fk_paper_to_density) or
exists(select 1 from my.paper_colors  where colors_id = new.fk_paper_color)
    then
        signal sqlstate '45000' set message_text = 'Нельзя вставить запись ';
    end if;
END
```

Рисунок 59. Код тригера

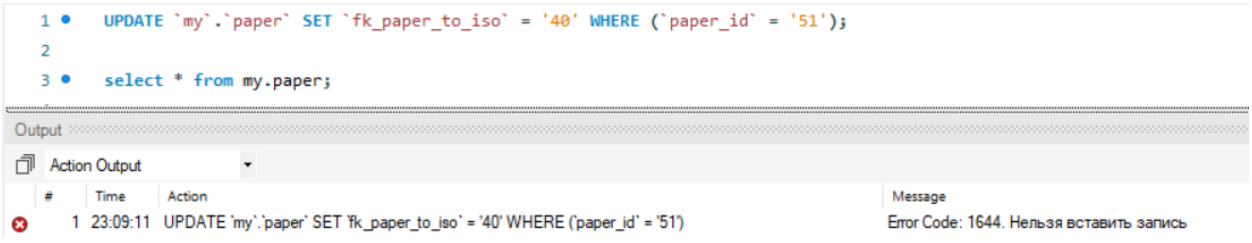

Рисунок 60. Код тригера

29. Тригер для Paper типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `paper_BEFORE_DELETE` BEFORE DELETE ON `paper` FOR EACH ROW BEGIN
if exists
(select 1 from my.services where fk_service_paper = old.paper_id)
    then
        signal sqlstate '45000' set message_text = 'Can not delete. There is a service uses this type';
    end if;
END
```

Рисунок 61. Код тригера

```
1 •  DELETE FROM `my`.`paper` WHERE (`paper_id` = '2');
2
```

Output

Action Output

#	Time	Action	Message
1	23:12:34	DELETE FROM `my`.`paper` WHERE (`paper_id` = '2')	Error Code: 1644. Can not delete. There is a service uses this type

Рисунок 62. Результат роботи тригера

30. Тригер для Paper_density типу **Before Update**

```
1 • CREATE DEFINER=`root`@`localhost` TRIGGER `paper_density_BEFORE_UPDATE` BEFORE UPDATE ON `paper_density` FOR EACH ROW BEGIN
2       if exists(select 1  from my.`paper` where fk_paper_to_density = old.density_id)
3       and old.density_id != new.density_id
4       then
5           signal sqlstate '45000' set message_text = 'can not update data ';
6       end if;
7   END
```

Рисунок 63. Код тригера

```
1 •  update my.paper_density
2    set density_id = 10
3    where density_id = 2
```

Output

Action Output

#	Time	Action	Message
1	11:27:23	update my.paper_density set density_id = 10 where density_id = 2	Error Code: 1644. can not update data

Рисунок 64. Результат роботи тригера

31. Тригер для Paper_density типу **Before Delete**

```
• CREATE DEFINER=`root`@`localhost` TRIGGER `paper_density_BEFORE_DELETE` BEFORE DELETE ON `paper_density` FOR EACH ROW BEGIN
      if exists(select 1  from my.`paper` where fk_paper_to_density = old.density_id)
      then
          signal sqlstate '45000' set message_text = 'can not delete data ';
      end if;
  END
```

Рисунок 65. Код тригера

```
1 •  delete from my.paper_density
2    where density_id = 2;
3 •  select * from my.paper_density;
```

Output

Action Output

#	Time	Action	Message
1	11:27:47	delete from my.paper_density where density_id = 2	Error Code: 1644. can not delete data

Рисунок 66. Результат роботи тригера

32. Тригер для Paper_brightness типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `paper_brightness_BEFORE_UPDATE` BEFORE UPDATE ON `paper_brightness` FOR EACH ROW BEGI
    if exists(select 1  from my.`paper` where fk_paper_to_bright = old.brightness_id)
    and old.brightness_id != new.brightness_id
    then
        signal sqlstate '45000' set message_text = 'can not update data ';
    end if;
END
```

Рисунок 67. Код тригера

```
update my.paper_brightness
set brightness_id = 49
where brightness_id = 1;
```

Output

Action Output

#	Time	Action	Message
1	23:24:13	update my.paper_brightness set brightness_id = 49 where brightness_id = 1	Error Code: 1644. can not update data

Рисунок 68. Результат роботи тригера

32. Тригер для Paper_brightness типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `paper_brightness_BEFORE_DELETE` BEFORE DELETE ON `paper_brightness` FOR EACH ROW BEGI
    if exists(select 1  from my.`paper` where fk_paper_to_bright = old.brightness_id)
    then
        signal sqlstate '45000' set message_text = 'can not delete data ';
    end if;
END
```

Рисунок 69. Код тригера

```
delete from my.paper_brightness
where brightness_id = 1;
```

Output

Action Output

#	Time	Action	Message
1	23:24:13	update my.paper_brightness set brightness_id = 49 where brightness_id = 1	Error Code: 1644. can not update data
2	23:24:56	delete from my.paper_brightness where brightness_id = 1	Error Code: 1644. can not delete data

Рисунок 70. Результат роботи тригера

33. Тригер для Paper_colors типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `paper_colors_BEFORE_UPDATE` BEFORE UPDATE ON `paper_colors` FOR EACH ROW BEGIN
    if exists(select 1  from my.`paper` where fk_paper_color = old.colors_id)
    and old.colors_id != new.colors_id
    then
        signal sqlstate '45000' set message_text = 'can not update data ';
    end if;
END
```

Рисунок 71. Код тригера

```
1  update my.paper_colors
2  set colors_id = 10
3  where colors_id = 2;
4  select * from my.paper_colors;
```

Output

Action Output

#	Time	Action	Message
1	10:56:03	update my.paper_colors set colors_id = 10 where colors_id = 2	Error Code: 1644. can not update data

Рисунок 72. Результат роботи тригера

34. Тригер для Paper_colors типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `paper_colors_BEFORE_DELETE` BEFORE DELETE ON `paper_colors` FOR EACH ROW BEGIN
if exists(select 1  from my.`paper` where fk_paper_color = old.colors_id)
    then
        signal sqlstate '45000' set message_text = 'can not delete data ';
    end if;
END
```

Рисунок 73. Код тригера

```
1  delete from my.paper_colors
2  where colors_id = 2;
3  select * from my.paper_colors;
```

Output

Action Output

#	Time	Action	Message
1	10:56:03	update my.paper_colors set colors_id = 10 where colors_id = 2	Error Code: 1644. can not update data
2	10:56:48	delete from my.paper_colors where colors_id = 2	Error Code: 1644. can not delete data

Рисунок 74. Результат роботи тригера

35. Тригер для Paper_covering типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `paper_covering_BEFORE_UPDATE` BEFORE UPDATE ON `paper_covering` FOR EACH ROW BEGIN
    if exists(select 1  from my.`paper` where fk_paper_covering = old.covering_id)
    and old.covering_id != new.covering_id
    then
        signal sqlstate '45000' set message_text = 'can not update data ';
    end if;
END
```

Рисунок 75. Код тригера

```
update my.paper_covering
set covering_id = 10
where covering_id = 2
```

```
Output
Action Output
#  Time      Action                                                                  Message
1  11:21:57  update my.paper_covering set covering_id = 10 where covering_id = 2    Error Code: 1644. can not update data
```

Рисунок 76. Результат роботи тригера

36. Тригер для Paper_covering типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `paper_covering_BEFORE_DELETE` BEFORE DELETE ON `paper_covering` FOR EACH ROW BEGIN
    if exists(select 1  from my.`paper` where fk_paper_covering = old.covering_id)
    then
        signal sqlstate '45000' set message_text = 'can not delete data ';
    end if;
END
```

Рисунок 77. Код тригера

```
delete from my.paper_covering
where covering_id = 2;
select * from my.paper_covering;
```

```
Output
Action Output
#  Time      Action                                                  Message
1  11:23:07  delete from my.paper_covering where covering_id = 2    Error Code: 1644. can not delete data
```

Рисунок 78. Результат роботи тригера

37. Тригер для Paper_iso типу **Before Update**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `paper_iso_BEFORE_UPDATE` BEFORE UPDATE ON `paper_iso` FOR EACH ROW BEGIN
    if exists(select 1  from my.`paper` where fk_paper_to_iso = old.iso_id)
    and old.iso_id != new.iso_id
    then
        signal sqlstate '45000' set message_text = 'can not update data ';
    end if;
END
```

Рисунок 79. Код тригера

```
update my.paper_iso
set iso_id = 10
where iso_id = 2
```

```
Output
Action Output
#  Time      Action                                          Message
1  11:30:23  update my.paper_iso set iso_id = 10 where iso_id = 2  Error Code: 1644. can not update data
```

Рисунок 80. Результат роботи тригера

38. Тригер для Paper_iso типу **Before Delete**

```
CREATE DEFINER=`root`@`localhost` TRIGGER `paper_iso_BEFORE_DELETE` BEFORE DELETE ON `paper_iso` FOR EACH ROW BEGIN
    if exists(select 1  from my.`paper` where fk_paper_to_iso = old.iso_id)

    then
        signal sqlstate '45000' set message_text = 'can not delete data ';
    end if;
END
```

Рисунок 81. Код тригера

```
delete from my.paper_iso
where iso_id = 2;
```

```
Output
Action Output
#  Time      Action                                  Message
1  11:30:50  delete from my.paper_iso where iso_id = 2  Error Code: 1644. can not delete data
```

Рисунок 82. Результат роботи тригера

Висновок:

Набуто практичних навичок з розробки тригерів для підтримки цілісності зв’язків, модифікації даних і забезпечення основних бізнес-процесів високонавантаженої інформаційної системи. Сформовано необхідні практичні уміння для аналізу плану виконання SQL-запитів за допомогою оператора EXPLAIN. Сформовано необхідні практичні уміння для створення тригерів, з урахуванням особливостей реалізації логіки роботи інтерфейсу високонавантаженої інформаційної системи зберігання даних.