

Практическое задание по теме “Транзакции, переменные, представления”

Оглавление

1	Практическое задание по теме “Транзакции, переменные, представления”	3
1.1	В базе данных shop и sample присутствуют одни и те же таблицы, учебной базы данных. Переместите запись id = 1 из таблицы shop.users в таблицу sample.users. Используйте транзакции.	3
1.2	Создайте представление, которое выводит название name товарной позиции из таблицы products и соответствующее название каталога name из таблицы catalogs.	4
1.3	(по желанию) Пусть имеется таблица с календарным полем created_at. В ней размещены разряженные календарные записи за август 2018 года '2018-08-01', '2016-08-04', '2018-08-16' и 2018-08-17. Составьте запрос, который выводит полный список дат за август, выставляя в соседнем поле значение 1, если дата присутствует в исходном таблице и 0, если она отсутствует.	5
1.4	(по желанию) Пусть имеется любая таблица с календарным полем created_at. Создайте запрос, который удаляет устаревшие записи из таблицы, оставляя только 5 самых свежих записей.	6
2	Практическое задание по теме “Администрирование MySQL” (эта тема изучается по вашему желанию).....	8
2.1	Создайте двух пользователей которые имеют доступ к базе данных shop. Первому пользователю shop_read должны быть доступны только запросы на чтение данных, второму пользователю shop — любые операции в пределах базы данных shop.	8
3	Практическое задание по теме “Хранимые процедуры и функции, триггеры”	9
3.1	Создайте хранимую функцию hello(), которая будет возвращать приветствие, в зависимости от текущего времени суток. С 6:00 до 12:00 функция должна возвращать фразу "Доброе утро", с 12:00 до 18:00 функция должна возвращать фразу "Добрый день", с 18:00 до 00:00 — "Добрый вечер", с 00:00 до 6:00 — "Доброй ночи".	9
3.2	В таблице products есть два текстовых поля: name с названием товара и description с его описанием. Допустимо присутствие обоих полей или одно из них. Ситуация, когда оба поля принимают неопределенное значение NULL неприемлема. Используя триггеры, добейтесь того, чтобы одно из этих полей или оба поля были заполнены. При попытке присвоить полям NULL-значение необходимо отменить операцию.....	10
3.3	(по желанию) Напишите хранимую функцию для вычисления произвольного числа Фибоначчи. Числами Фибоначчи называется последовательность в которой	

число равно сумме двух предыдущих чисел. Вызов функции FIBONACCI(10) должен
возвращать число 55.11

1 Практическое задание по теме “Транзакции, переменные, представления”

- 1.1 В базе данных shop и sample присутствуют одни и те же таблицы, учебной базы данных. Переместите запись id = 1 из таблицы shop.users в таблицу sample.users. Используйте транзакции.

```
USE shop;  
use sample;  
SELECT * FROM users;
```

```
START TRANSACTION;  
INSERT INTO sample.users( name) SELECT name FROM shop.users WHERE  
users.id = 1 ;  
DELETE FROM shop.users where id=1;  
COMMIT;
```

- 1.2 Создайте представление, которое выводит название name товарной позиции из таблицы products и соответствующее название каталога name из таблицы catalogs.

```
USE shop;
```

```
CREATE OR REPLACE VIEW prod1
AS SELECT products.name AS products , catalogs.name AS catalog
   FROM products
  LEFT JOIN catalogs
    ON products.catalog_id = catalogs.id;

SELECT * FROM prod1;
```

- 1.3 (по желанию) Пусть имеется таблица с календарным полем `created_at`. В ней размещены разряженные календарные записи за август 2018 года '2018-08-01', '2018-08-04', '2018-08-16' и '2018-08-17'. Составьте запрос, который выводит полный список дат за август, выставя в соседнем поле значение 1, если дата присутствует в исходном таблице и 0, если она отсутствует.

```
use example;
```

```
CREATE TABLE dates (  
  id SERIAL PRIMARY KEY,  
  date1 DATE COMMENT "ДАТА"  
) COMMENT = "Dates of october";
```

```
INSERT INTO dates(id, date1) VALUES  
(null, '2018-08-01'),  
(null, '2018-08-04'),  
(null, '2018-08-16'),  
(null, '2018-08-17');
```

```
DROP TABLE IF EXISTS last_days;  
CREATE TEMPORARY TABLE last_days  
(num INT);
```

```
INSERT INTO last_days VALUES  
(1), (2), (3), (4), (5), (6), (7), (8), (9), (10),  
(11), (12), (13), (14), (15), (16), (17), (18), (19), (20),  
(21), (22), (23), (24), (25), (26), (27), (28), (29), (30);
```

```
SELECT STR_TO_DATE(concat('2018-08-',num), '%Y-%m-%d') as date  
FROM last_days;
```

	date
▶	2018-08-01
	2018-08-02
	2018-08-03
	2018-08-04
	2018-08-05

```

SELECT STR_TO_DATE(concat('2018-08-',num), '%Y-%m-%d') as date,
NOT(ISNULL(dates.date1))
FROM last_days
LEFT JOIN
dates
ON STR_TO_DATE(concat('2018-08-',num), '%Y-%m-%d') = dates.date1 ;

```

	date	NOT(ISNULL(dates.date1))
▶	2018-08-01	1
	2018-08-02	0
	2018-08-03	0
	2018-08-04	1
	2018-08-05	0

- 1.4 (по желанию) Пусть имеется любая таблица с календарным полем `created_at`. Создайте запрос, который удаляет устаревшие записи из таблицы, оставляя только 5 самых свежих записей.

```

use example;
DROP TABLE IF EXISTS old_dates;
CREATE TABLE old_dates (
id SERIAL PRIMARY KEY,
name VARCHAR(255),
created_at DATE NOT NULL);

```

```

INSERT INTO old_dates VALUES
(null, '1', '2010-11-01'),
(null, '2', '2012-11-01'),
(null, '3', '2013-11-01'),
(null, '4', '2014-11-01'),
(null, '5', '2015-11-01'),
(null, '6', '2016-11-01'),
(null, '7', '2017-11-01'),
(null, '8', '2018-11-01');

```

```

SELECT * FROM old_dates;

```

	id	name	created_at
▶	1	1	2010-11-01
	2	2	2012-11-01
	3	3	2013-11-01
	4	4	2014-11-01
	5	5	2015-11-01
	6	6	2016-11-01
	7	7	2017-11-01
	8	8	2018-11-01
▲	NULL	NULL	NULL

Выводит 6-ю строчку с датой:

```
SELECT created_at from old_dates  
ORDER BY created_at DESC  
limit 5,1;
```

Вы водит строки с датой старше чем 6-я (для удаления)

```
SELECT * FROM old_dates  
join (SELECT created_at from old_dates  
ORDER BY created_at DESC  
limit 5,1) as sixth_post  
ON old_dates.created_at <= sixth_post.created_at;
```

	id	name	created_at	created_at
▶	1	1	2010-11-01	2013-11-01
	2	2	2012-11-01	2013-11-01
	3	3	2013-11-01	2013-11-01

Удаляем строки

```
DELETE old_dates FROM old_dates  
join (SELECT created_at from old_dates  
ORDER BY created_at DESC  
limit 5,1) as sixth_post  
ON old_dates.created_at <= sixth_post.created_at;
```

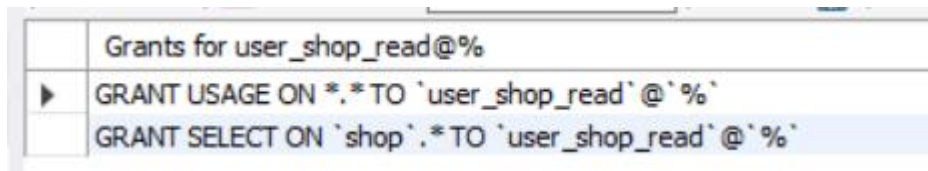
Результат:

	id	name	created_at
▶	4	4	2014-11-01
	5	5	2015-11-01
	6	6	2016-11-01
	7	7	2017-11-01
	8	8	2018-11-01
•	NULL	NULL	NULL

2 Практическое задание по теме “Администрирование MySQL” (эта тема изучается по вашему желанию)

- 2.1 Создайте двух пользователей которые имеют доступ к базе данных shop. Первому пользователю shop_read должны быть доступны только запросы на чтение данных, второму пользователю shop — любые операции в пределах базы данных shop.

```
DROP USER IF EXISTS user_shop_read;
CREATE USER user_shop_read IDENTIFIED WITH sha256_password BY '123';
REVOKE SELECT, INSERT, UPDATE, DELETE, CREATE, ALTER, DROP ON *.* FROM
user_shop_read ;
GRANT SELECT ON shop.* TO user_shop_read;
SHOW GRANTS FOR user_shop_read;
```



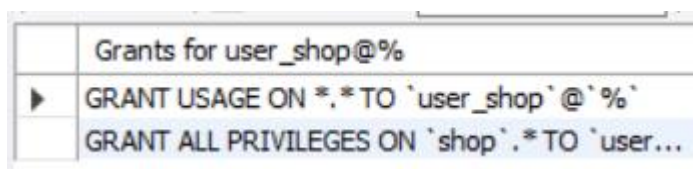
	Grants for user_shop_read@%
▶	GRANT USAGE ON *.* TO `user_shop_read`@`%`
	GRANT SELECT ON `shop`.* TO `user_shop_read`@`%`

PS: команда

```
REVOKE ALL ON contacts FROM user_shop_read;
```

не работает

```
CREATE USER user_shop IDENTIFIED WITH sha256_password BY '123';
GRANT ALL ON shop.* TO user_shop;
SHOW GRANTS FOR user_shop;
```



	Grants for user_shop@%
▶	GRANT USAGE ON *.* TO `user_shop`@`%`
	GRANT ALL PRIVILEGES ON `shop`.* TO `user...

3 Практическое задание по теме “Хранимые процедуры и функции, триггеры”

- 3.1 Создайте хранимую функцию `hello()`, которая будет возвращать приветствие, в зависимости от текущего времени суток. С 6:00 до 12:00 функция должна возвращать фразу "Доброе утро", с 12:00 до 18:00 функция должна возвращать фразу "Добрый день", с 18:00 до 00:00 — "Добрый вечер", с 00:00 до 6:00 — "Доброй ночи".

```
USE example;
DROP FUNCTION IF EXISTS hello;
DELIMITER //
CREATE FUNCTION hello()
RETURNS tinytext NO SQL
BEGIN
    DECLARE hour INT;
    SET hour= HOUR(NOW());
    CASE
        WHEN hour BETWEEN 0 AND 5 THEN
            RETURN "Good Night";
        WHEN hour BETWEEN 6 AND 11 THEN
            RETURN "Good Morning";
        WHEN hour BETWEEN 12 AND 17 THEN
            RETURN "Good Day";
        WHEN hour BETWEEN 18 AND 23 THEN
            RETURN "Good Evening";
    END CASE;
END//
DELIMITER ;
```

```
SELECT NOW(), hello();
```

	NOW()	hello()
▶	2022-09-19 14:41:39	Good Day

3.2 В таблице products есть два текстовых поля: name с названием товара и description с его описанием. Допустимо присутствие обоих полей или одно из них. Ситуация, когда оба поля принимают неопределенное значение NULL неприемлема. Используя триггеры, добейтесь того, чтобы одно из этих полей или оба поля были заполнены. При попытке присвоить полям NULL-значение необходимо отменить операцию.

```
USE shop;
```

```
DELIMITER //
```

```
CREATE TRIGGER validate_name_description_insert BEFORE INSERT ON  
products
```

```
FOR EACH ROW
```

```
BEGIN
```

```
IF NEW.name IS NULL AND NEW.description IS NULL THEN
```

```
    SIGNAL SQLSTATE '45000'
```

```
    SET MESSAGE_TEXT = 'BOTH NAME AND DESCRIPTION ARE NULL';
```

```
END IF;
```

```
END//
```

```
INSERT INTO products
```

```
(name, description, price, catalog_id)
```

```
VALUES
```

```
(NULL, NULL, 9360, 2);
```

✖ 126 14:55:36 INSERT INTO products (name, description, price, catalog_id) VALUES (NULL, NULL, 9360, 2); Error Code: 1644. BOTH NAME AND DESCRIPTION ARE NULL. 0.000 sec

```
INSERT INTO products
```

```
(name, description, price, catalog_id)
```

```
VALUES
```

```
(NULL, 'HDMI, SATA3 PCI EXPRESS 3.0,, USB3.1', 9360, 2);
```

```
SELECT * FROM products;
```

Продукт вставлен:

	id	name	description	price
▶	1	Intel Core i3-8100	Процессор для настольных персональных ко...	7890.00
	2	Intel Core i5-7400	Процессор для настольных персональных ко...	12700.00
	3	AMD FX-8320E	Процессор для настольных персональных ко...	4780.00
	4	AMD FX-8320	Процессор для настольных персональных ко...	7120.00
	5	ASUS ROG MAXIMUS X HERO	Материнская плата ASUS ROG MAXIMUS X HE...	19310.00
	6	Gigabyte H310M S2H	Материнская плата Gigabyte H310M S2H, H31...	4790.00
	7	MSI B250M GAMING PRO	Материнская плата MSI B250M GAMING PRO, ...	5060.00
	8	NULL	HDMI, SATA3 PCI EXPRESS 3.0,, USB3.1	9360.00

- 3.3 (по желанию) Напишите хранимую функцию для вычисления произвольного числа Фибоначчи. Числами Фибоначчи называется последовательность в которой число равно сумме двух предыдущих чисел. Вызов функции FIBONACCI(10) должен возвращать число 55.

```
USE example;

DROP FUNCTION IF EXISTS fibonacci;

DELIMITER //
CREATE FUNCTION fibonacci (step INT)
RETURNS INT DETERMINISTIC
BEGIN
    DECLARE fib1, fib2, fib3, step_now INT;
    SET step_now = 1;
    SET fib1 = 0;
    SET fib2 = 1;
    cycle: LOOP
        SET step_now = step_now + 1;
        SET fib3 = fib1 + fib2;
        IF step_now = step THEN
            RETURN fib3;
        END IF;
        SET fib1 = fib2;
        SET fib2 = fib3;
    END LOOP cycle;
END//

DELIMITER ;

SELECT fibonacci(10);
```

	fibonacci(10)
▶	55