

Лабораторийн ажил 7. ӨС-гийн боловсруулалт-Сторед процедур, курсор ашиглах

Лабораторийн ажлын зорилго

- сторед процедур бичиж сурах
- сторед процедур руу параметр дамжуулах, буцах утгыг параметрээр дамжуулах
- Өгөгдөл боловсруулахад хувьсагч, давталт, нөхцөл шалгах үйлдлүүд ашиглах
- Сторед процедур дотор курсор зарлах
- Давталт ашиглаж бүх өгөгдлийг уншиж боловсруулах

Даалгавар-1

Оюутан бүр Бие даалтын ажлаар сонгосон сэдвийн дагуу байгуулсан ӨС схемийг ашиглаж ӨС үүсэх скрипыг ажиллуулж , ӨС-г шинээр үүсгэх.

- Бие Даалт №1,2 дээр үүсгэсэн (Лавлах хүснэгтүүдэд бүх бичлэгийг оруулсан байх - Бодит утгууд оруулна) скрипт файлыг хуулж Лаборатори №7 файл үүсгэ
- Лаборатори №7 скрипт файлыг ажиллуулж ӨС-г үүсгэ

Даалгавар-2

БД-н Функциональ шаардлага дээр тодорхойлсон өгөдөл боловсруулах **ОБЪЕКТЫН НЭР ТООЦООЛОХ** функцүүдийн Кверийг **КУРСОР ашигласан СТОРЕД ПРОЦЕДУР** ашиглаж бүтээх

- Сторед процедурын параметрийг IN, OUT, INOUT ашиглаж өгөгдөл оруулах
- Локал хувьсагч ашиглах
- Курсор зарлах
- Курсор төгсгөлдөө хүрсэнг шалгах CONTINUE HANDLER зарлаж ашиглах
- Давталт ашиглаж нийт өгөгдлийг уншиж боловсруулах
- **Оюутан тус бүр дор хаяж НЭГ ширхэг тооцоолох сторед процедур бүтээж лабораторийг тооцно**
- **Баг тус бүр дор хаяж ХОЁР ширхэг тооцоолох сторед процедуруудыг бүтээж БД-ыг тооцно**

Хугацаа

Лабораторийн цаг дээр үзүүлнэ.

Гүйцэтгэж хамгаалах шаардлага

- Лабораторийн ажил №7 скрипт файлаас ӨС-ийн үүсгэх
- Даалгавар №2 -оор курсор ашигласан сторед процедур бүтээсэн кверийг тайлбарлах
- Бүтээсэн сторед процедураа ашиглаж тооцоолол хийж тайлбарлах

Нэмэлт материал

Онолын хэсэг, жишээ

Курсор

Курсор нь query-ний үр дүнд үүссэн бичлэгийн олонлогийн бичлэг тус бүрт давталт ашиглан хандаж тодорхой үйлдлийг гүйцэтгэх боломжийг олгодог..

	buy_id	buy_bok_id	buy_qty	buy_ord_id	bok_id	bok_name	bok_price	bok_status	
▶	1	1	2	1	1	Twilight	100.00	1	→ Үйлдэл гүйцэтгэх
	3	2	2	1	2	Davinci Code	10.00	2	→ Үйлдэл гүйцэтгэх
	4	1	1	1	1	Twilight	100.00	1	→ Үйлдэл гүйцэтгэх
	5	2	1	1	2	Davinci Code	10.00	2	→ Үйлдэл гүйцэтгэх

Амьдралын цикл нь дараах байдлаар тодорхойлогдоно:

1. **DECLARE CURSOR** илэрхийллээр зарлагдах бөгөөд SQL серверийн санах ойд курсорыг үүсгэнэ.
2. **OPEN CURSOR** илэрхийллээр курсорыг идэвхижүүлнэ. Ө/х: курсорыг хамгийн эхний бичлэгтэй холбоно.
3. **FETCH** түлхүүр үгээр курсороор дамжуулж өгөгдөлд хандана. MOVE – заасан бичлэг рүү шилжих
4. **CLOSE CURSOR** илэрхийллээр курсорыг идэвхигүй болгоно. Ингэснээр нэмэлт мөрүүдтэй холбох боломжгүй болно

Синтакс:

DECLARE курсор_нэр CURSOR FOR SELECT илэрхийлэл;

OPEN курсор_нэр;

FETCH курсор_нэр INTO хувьсагчийн жагсаалт;

CLOSE курсор_нэр;

Алдаа баригч -CONTINUE ... HANDLER:

Курсор дахь бичлэгүүдийг хэзээ унших вэ гэдгийг давталт дотроо зааж өгнө.

Үүний тулд **NOT FOUND** алдаа баригчийг зарлана.

Синтакс:

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

Бичлэгийг уншиж дууссан үед done = 1 болох ба давталтыг дуусгах ёстой.

NOT FOUND –ийн оронд **SQLSTATE** командыг ашиглаж болно.

SQLSTATE код нь SQL алдааг мэдээллэнэ. Алдаа нь тоо болон ASCII том үсгүүд агуулсан 5 тэмдэгтээс бүрдэнэ. **SQLSTATE** утгын эхний 2 тэмдэгт нь алдааны ангилал, сүүлийн 3 тэмдэгт нь дэд ангиллыг илэрхийлнэ. Жишээ нь, **SQLSTATE '02000'** алдаа нь бичлэг олдоогүй үед үүснэ.

Жишээ:

```

DELIMITER //

CREATE PROCEDURE sp_process_orders()

BEGIN

    DECLARE done BOOLEAN DEFAULT 0;
    DECLARE o INT;
    DECLARE ordernumbers CURSOR
    FOR
    SELECT odr_num FROM orders;

    DECLARE CONTINUE HANDLER
    FOR SQLSTATE '02000' SET done=1;

    OPEN ordernumbers;

    get_ord_number: LOOP
        FETCH ordernumbers INTO o;
        IF done = 1 THEN
            LEAVE get_ord_number;
        END IF;
        SELECT o;
    END LOOP get_ord_number;

    CLOSE ordernumbers;

END //

DELIMITER ;

```

Локаль хувьсагч

Курсор зарлалт

NOT FOUND ашиглаж болно

continue handler зарлах

курсор нээх

Нэг өгөгдөл унших

бичлэгүүдийг давталтаар унших

Курсороо хаах

Сторед процедурыг дуудахад

CALL sp_processorders();

```

> call sp_process_orders
Result set 1
***** 1. row
o: 1
1 rows in set

Result set 2
***** 1. row
o: 2
1 rows in set

```

odr_num	odr_cust_num	odr_unit_price	odr_quantity	odr_total
1	1	10.00	NULL	NULL
2	2	10.00	NULL	NULL
NULL	NULL	NULL	NULL	NULL

О хувьсагч

Үр дүнд products хүснэгтийн мөрийг тоолж гарах тоог total_products хувьсагчдад утга болгон оноож байна.

Жишээ: Алдаа баригч(handler)-ийн оронд COUNT ашиглаж болно.Бүтэц:

```
DELIMITER //
CREATE PROCEDURE sp_processorders2 ()
BEGIN
    -- Локаль хувьсагч зарлана
    DECLARE count_order INT DEFAULT 0;
    DECLARE o INT;
    -- Курсороо зарлана
    DECLARE ordernumbers CURSOR
    FOR
    SELECT odr_numbr FROM orders;
    -- Бичлэгийн тоог авна
    SELECT COUNT(*) INTO count_order FROM orders;
    -- Курсороо нээнэ
    OPEN ordernumbers;
    -- Бүх бичлэгүүдийн хувьд давтана
    REPEAT
    -- Захиалгын дугаарыг авна
    FETCH ordernumbers INTO o;
    SELECT o;
    SET count_order=count_order-1;
    -- Давталт дуусна
    UNTIL count_order=0 END REPEAT;
    -- Курсороо хаана
    CLOSE ordernumbers;
END //
DELIMITER ;
```

Процедураа дуудах:

CALL sp_processorders2 ()

	odr_numbr	odr_cust_numbr	odr_unit_price	odr_quantity	odr_total
▶	1	1	10.00	NULL	NULL
	2	2	10.00	NULL	NULL
*	NULL	NULL	NULL	NULL	NULL

```
> call sp_process_orders2

Result set 1
***** 1. row *****
o: 1
1 rows in set

Result set 2
***** 1. row *****
o: 2
1 rows in set
```

Үр дүн:

	0
▶	2

Жишээ:

Бүх номны хувьд уг ном борлуулагдсан эсэхийг шалгаад, хэрэв борлуулагдаагүй ном байвал **infologs** хүснэгтэнд мэдээлэл оруулна:

```
DELIMITER //
```

```
DROP PROCEDURE IF EXISTS sp_no_buy$$
```

```
CREATE PROCEDURE sp_no_buy()
```

```
BEGIN
```

```
    DECLARE no_more_products, quantity_bought INT DEFAULT 0;
```

```
    DECLARE bookId VARCHAR(255);
```

```
    DECLARE cur_book_id CURSOR FOR SELECT bok_id FROM book;
```

```
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET
```

```
                                no_more_products = 1;
```

```
CREATE TABLE infologs (
```

```
inf_id INT NOT NULL AUTO_INCREMENT,
```

```
inf_bok_id INT,
```

```
inf_msg VARCHAR(255) NOT NULL,
```

```
PRIMARY KEY (inf_id)
```

```
);
```

```
OPEN cur_book_id;
```

```
-- эхний номны мэдээллийг авах
```

```
FETCH cur_book_id INTO bookId;
```

```
REPEAT
```

```
-- тухайн номыг хэдэн удаа авсаныг тоолох
```

```
SELECT COUNT(*) INTO quantity_bought
```

```
FROM buy
```

```
WHERE buy_bok_id = bookId;
```

```
-- тухайн ном ганц ч борлуулагдаагүй бол мэдээллийг нь лог
```

```
хүснэгтэнд бичих
```

```
IF quantity_bought = 0 THEN
```

```
INSERT INTO infologs(inf_bok_id, inf_msg)
```

```
VALUES (bookId, ' энэ ном ганц ч борлуулагдаагүй байна');
```

```
END IF;
```

```
-- дараагийн номны мэдээлэлтэй холбогдох
```

```
FETCH cur_book_id INTO bookId;
```

```
UNTIL no_more_products = 1
```

```
END REPEAT;
```

```
CLOSE cur_book_id;
```

```
SELECT * FROM infologs;
```

```
DROP TABLE infologs;
```

```
END//
```

```
DELIMITER ;
```

Процедураа дуудах:

Өгөгдөл

```

1 SELECT *
2 FROM book LEFT OUTER JOIN buy
3 ON bok_id= buy_bok_id;

```

100%
23:3

Filter:
Export:

	bok_id	bok_name	buy_id	buy_ord_id	buy_bok_id
▶ 1	1	Twilight	1	1	1
1	1	Twilight	4	1	1
2	2	Da Vinci Code	2	2	2
2	2	Da Vinci Code	3	1	2
2	2	Da Vinci Code	5	1	2
3	3	Very complicated book	NULL	NULL	NULL
4	4	strange book	NULL	NULL	NULL

call sp_no_buy;

Үр дүн

inf_id	inf_bok_id	inf_msg
1	3	This book has never been bought
2	4	This book has never been bought