

## **B. Sc. Engg. 3rd year 1st Semester**

DBMS LAB

### DISCUSSION-3

#### 1. Stored Procedure

Stored Procedures are database objects where multiple SQL statements can be executed as a batch. Stored procedures once created stays in the database and can be executed from client side.

The following stored procedure shows authors name of a given title\_id

```
CREATE PROC sp_showTitleAndAuthor
AS
BEGIN
SELECT "Authors Last Name"=au_lname FROM authors where au_id in (select au_id from
titleauthor where title_id='BU1032')
END
```

To execute the just created Stored Procedure the command is

```
EXEC sp_showTitleAndAuthor
```

To modify an existing stored procedure use the following statements

```
ALTER PROC sp_showTitleAndAuthor
AS
BEGIN
---
```

```
END
```

To delete the stored procedure from the database

```
DROP PROC sp_showTitleAndAuthor
```

#### 2. Parameterized Stored procedure

Like function arguments Stored Procedures can accept values when being executed and can also return values.

Example: Modifying the procedure created in 1 that accepts an title\_id and shows the corresponding author name

```
ALTER PROC sp_showTitleAndAuthor @titleid char(15)
AS
BEGIN
SELECT "Authors Last Name"=au_lname FROM authors where au_id in (select au_id from
```

```
titleauthor where title_id=@titleid)
END
```

### 3. Stored procedures with decision making/ looping constructs

The following procedure can be used to increase the price of a particular book by 10% but on the condition that the new price does not cross \$20

```
CREATE PROC sp_updatePrice @titleid char(15)
AS
BEGIN
    DECLARE @price MONEY
    SELECT @price=price from TITLES WHERE title_id=@titleid
    set @price=@price+0.1*@price
    IF @price<=20
        UPDATE titles SET price= @price WHERE title_id=@titleid
END
```

```
EXEC sp_updatePrice 'BU7832'
```

---

### Assignments

Using the tables created in the last class (i.e., CustomerAndSuppliers, Items, Transactions) perform the following tasks

#### Task 1:

Write a stored procedure that prints out item categories, total number of items available and average price of that category in the following format.

<u>Category</u>	<u>Total number of items</u>	<u>Average Price</u>
-----------------	------------------------------	----------------------

#### Task: 2

Write a stored procedure that

- a) accepts as two inputs, i.e., i) category name and ii) price value
- b) And shows the item details under that category that are cheaper than the accepted price value

#### Task 3:

Write a stored procedure that

- a) Accepts as input i) category name and ii) desired average price value
- b) And increase the price of each item under that category by 10% until the new average price crosses the desired average price value.