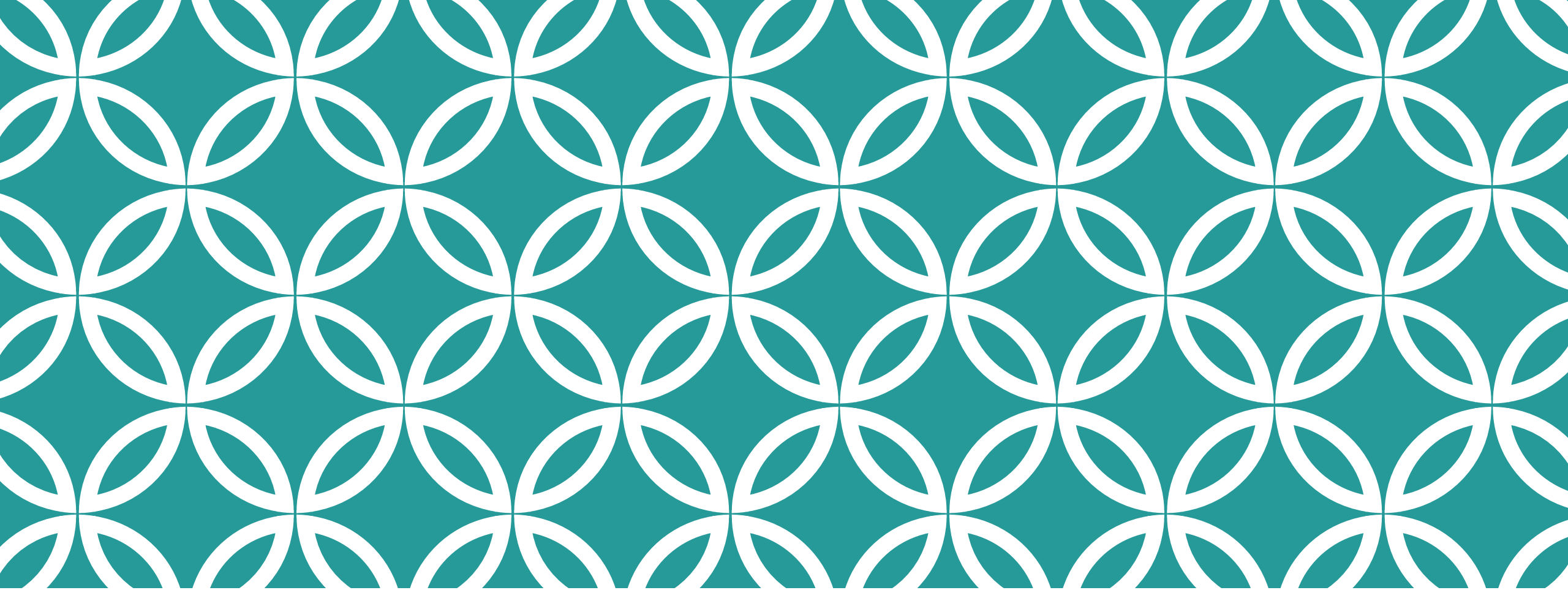

CHAPTER 19: STORED PROCEDURES

2023

CHAPTER 19: STORED PROCEDURES

Contents:

1. Introduction to stored procedures.
2. Reasons to use stored procedures.
3. Example of procedures.

STORED PROCEDURES

A stored procedure is a group of SQL statements that has been created and then stored in that database

Stored procedures are functions/Query tasks that you can save in order to use later.

REASONS TO USE STORED PROCEDURES

- Efficient execution of common tasks
- Reuse tasks that you can use in several ways
- To make changes, without changing the original data

STORED PROCEDURES

```
CREATE PROCEDURE procedure_name
AS
BEGIN
   Query;
END;
```

The command to execute:

```
Exec procedurename;
```

STORED PROCEDURES

```
CREATE PROCEDURE GetProductDesc
AS
BEGIN
SET NOCOUNT ON
SELECT P.ProductID, P.ProductName, PD.ProductDescription  FROM Product P
INNER JOIN ProductDescription PD ON P.ProductID=PD.ProductID
END
```

Procedure name

Inner Query, gets the fields specified each time the procedure runs

**Always ensure you close the procedure*

STORED PROCEDURES

At times you would like to create a procedure that brings back data that is for a different criterion or that has parameters this can be done by adding a tag name:

```
CREATE PROCEDURE procedure_name @Lastname datatype(Len)
AS
BEGIN
Query; (remember to add the tag to your query)
END;
```

STORED PROCEDURES

```
CREATE PROCEDURE dbo.uspGetAddress @City nvarchar(30)
AS
BEGIN
SELECT *
FROM Person. Address
WHERE City = @City
END
```

Always give the data type and Length of the input when creating a parameter-based procedure

To execute : EXEC dbo.uspGetAddress @City = 'New York'

Practice STORED PROCEDURES

1. Create a stored procedure call listing250, that will extract all listings that are greater than 250.
2. Create a stored procedure call listingX, that will extract all listings that are greater than the value X. Where the value of X will be passed on/tagged in the procedure.