

# You've got the power

INTRODUCTION TO SQL SERVER



John MacKintosh  
Instructor

# CRUD operations

## CREATE

- Databases, Tables or views
- Users, permissions, and security groups

## READ

- Example: `SELECT` statements

## UPDATE

- Amend existing database records

## DELETE

# CREATE

- `CREATE TABLE` `unique table name`
- `(column name, data type, size)`

```
CREATE TABLE test_table(  
  test_date date,  
  test_name varchar(20),  
  test_int int  
)
```

# A few considerations when creating a table

- Table and column names
- Type of data each column will store
- Size or amount of data stored in the column

# Data types

## Dates:

- `date ( YYYY-MM-DD )`, `datetime ( YYYY-MM-DD hh:mm:ss )`
- `time`

## Numeric:

- `integer`, `decimal`, `float`
- `bit ( 1 = TRUE , 0 = FALSE . Also accepts NULL values)`

## Strings:

- `char` , `varchar` , `nvarchar`

# Let's create some tables!

INTRODUCTION TO SQL SERVER

# Insert, Update, Delete

INTRODUCTION TO SQL SERVER



**John MacKintosh**  
Instructor

# INSERT

```
INSERT INTO table_name
```

```
INSERT INTO table_name (col1, col2, col3)
```

```
INSERT INTO table_name (col1, col2, col3)  
VALUES  
('value1', 'value2', value3)
```



# INSERT SELECT

```
INSERT INTO table_name (col1, col2, col3)
SELECT
    column1,
    column2,
    column3
FROM other_table
WHERE
    -- conditions apply
```

- Don't use `SELECT *`
- Be specific in case table structure changes

# UPDATE

```
UPDATE table
SET column = value,
WHERE
  -- Condition(s);
```

- Don't forget the `WHERE` clause!

```
UPDATE table
SET
  column1 = value1,
  column2 = value2
WHERE
  -- Condition(s);
```

# DELETE

```
DELETE  
FROM table  
WHERE  
    -- Conditions
```

- Test beforehand!

```
TRUNCATE TABLE table_name
```

- Clears the entire table at once



# Let's INSERT, UPDATE, and DELETE!

INTRODUCTION TO SQL SERVER

# Declare yourself

INTRODUCTION TO SQL SERVER



John MacKintosh  
Instructor

# Variables

```
SELECT *  
FROM artist  
WHERE name = 'AC/DC';
```

Now change the query for another artist:

```
SELECT *  
FROM artist  
WHERE name = 'U2';
```

To avoid repetition, create a variable:

```
SELECT *  
FROM artist  
WHERE name = @my_artist;
```

# DECLARE

```
DECLARE @
```

Integer variable:

```
DECLARE @test_int INT
```

Varchar variable:

```
DECLARE @my_artist VARCHAR(100)
```

# SET

Integer variable:

```
DECLARE @test_int INT
```

```
SET @test_int = 5
```

Assign value to @my\_artist :

```
DECLARE @my_artist varchar(100)
```

```
SET @my_artist = 'AC/DC'
```



```
DECLARE @my_artist varchar(100)
DECLARE @my_album varchar(300);

SET @my_artist = 'AC/DC'
SET @my_album = 'Let There Be Rock' ;

SELECT --
FROM --
WHERE artist = @my_artist
AND album = @my_album;
```

```
DECLARE @my_artist varchar(100)
DECLARE @my_album varchar(300);

SET @my_artist = 'U2'
SET @my_album = 'Pop' ;

SELECT --
FROM --
WHERE artist = @my_artist
AND album = @my_album;
```

# Temporary tables

```
SELECT
  col1,
  col2,
  col3 INTO #my_temp_table
FROM my_existing_table
WHERE
  -- Conditions
```

- `#my_temp_table` exists until connection or session ends

```
-- Remove table manually
DROP TABLE #my_temp_table
```

# Let's declare some variables!

INTRODUCTION TO SQL SERVER

# Congratulations!

INTRODUCTION TO SQL SERVER



John MacKintosh  
Instructor

# What we learned...

- Selecting: `SELECT`
- Ordering: `ORDER BY`
- Filtering: `WHERE` and `HAVING`
- Aggregating: `SUM` , `COUNT` , `MIN` , `MAX` and `AVG`
- Text manipulation: `LEFT` , `RIGHT` , `LEN` and `SUBSTRING`

# What we learned (II)...

- `GROUP BY`
- `INNER JOIN` , `LEFT JOIN` , `RIGHT JOIN`
- `UNION` and `UNION ALL`
- Create, Read, Update and Delete (CRUD)
- Variables
- Temporary tables

# Next Steps

- Intermediate SQL Server
- Joining Data in SQL

# Congratulations!

INTRODUCTION TO SQL SERVER