Accessing the Data

- Basic Query Structure
    - Example SELECT statements
    - Logical Query Processing Phases
        - Keyed in order:
            - SELECT
            - FROM
            - WHERE
            - GROUP BY
            - HAVING
            - ORDER BY
        - Logical Query Processing Order
            - FROM
            - WHERE
            - GROUP BY
            - HAVING
            - SELECT
            - ORDER BY
    - FROM Clause
        - Specifies what table(s) you want to get data from.
        - Includes JOINS
    - WHERE Clause
        - Filters the data to return only what you asked for
        - (Any row that returned false are not shown)
    - GROUP BY Clause
        - Groups results for each distinct combination of values in the grouped elements
        - This is powerful but tricky and I don't think we will use it much in a web app
    - HAVING Clause
        - Filters data after it has been grouped.
        - Don't think we will use this much in a web app
    - SELECT Clause
        - This has 2 phases.
            - Step 1
                - Evaluates the expressions in the SELECT list and produces the result set
            - Step 2
                - If you use the DISTINCT clause, this step removes duplicate records
    - ORDER BY Clause
        - SQL Statement have no predetermined order. It may look like the data returns in the same order every time, but unless you order the data, you cannot guarantee it!
        - Seen this happen myself, older program.

- Data Types
  - Exact Numeric
    - INT
    - NUMERIC
      - Number with exact precision (decimal points)
  - Character Strings
    - CHAR
      - Fixed length
    - VARCHAR
      - Variable length
  - Unicode Character Strings
    - NCHAR
    - NVARCHAR
  - Approximate Numeric
    - FLOAT
      - Scientific notation
    - REAL
  - Binary String
    - BINARY
    - VARBINARY
  - Date and Time
    - DATE
      - Year, month, day
    - TIME
      - Hour, minute, seconds
    - TIMESTAMP
      - Year, month, day, hour, minute, seconds
- CAST
  - Converts data from one type into another type if possible.
- NULL
  - NULL Values represent missing unknown data
  - NULL != NULL
    - You cannot compare a NULL to a NULL without help.
  - IS NULL operator
  - NULL != 0
    - NULL is NOT the same as 0
    - Although sometimes you can compare it and get a useable results, don't do this.
- Filtering data – WHERE Clause Operators
  - =
  - <>
    - Sometimes != works, but not always, better to use <>
  - >, <, >=, <=
  - BETWEEN
  - LIKE
  - IN
  - AND
  - OR

Creating and Altering Tables

- You may not create a lot of tables as a Web Dev, but knowing what you can setup in the table when creating it, will help shape your SQL Queries.
- Two basic ways of creating a table
    - CREATE TABLE
    - SELECT INTO
        - Creates a table based on the result set of a SELECT Query, inheriting most of the column properties
- CREATE TABLE syntax
    - Column, Type, Special, Null, default.
- IDENTITY works differently in different SQL Engines!
- ALTER TABLE
    - Add or remove a column
    - Change the data type of a column
    - Change NULL or NOT NULL
    - Add or remove a CONSTRAINT
- CONSTRAINT – Data Integrity
    - It looks like a lot of the data validation is done in the web application form, but that's not always a good idea. If data validation is only done when you are entering a form, another user may be able to INSERT data directly into the database, by passing all data validation, causing data corruption or unclean data. It's good to be able to recognize constraints and formulate your application to use both form validation and database constraint logic.
    - Can be set during CREATE TABLE statement
    - Can be added later using the ALTER TABLE statement
    - CONSTRAINT Types
        - PRIMARY KEY
            - Every table in a relational database needs a way of distinguishing each row from all of the others. This is the PRIMARY KEY's job. Each value in the PRIMARY KEY column must be unique.
        - UNIQUE
            - Enforces all data in each row to be unique.
        - FOREIGN KEY
            - A FOREIGN KEY is a column in one table that serves as a link to look up data in another table.
- Example queries with PK and FK

Manipulating Data

- INSERT
  - INSERT VALUES
    - You can insert one or more rows into the target table
  - INSERT SELECT
    - You can insert the results set from a SELECT statement into a table
  - SELECT INTO
    - You can create a table from the result set of a SELECT statement, and some table properties will be inherited, but not all. (CONSTRAINTS are not inherited).
- UPDATE
  - ALWAYS use a WHERE clause!
  - UPDATE *table* SET *column* <operator> value
- DELETE
  - ALWAYS use a WHERE clause!
  - DELETE allows you to delete specific rows from a table
  - DELETE FROM *table* WHERE *something*
- TRUNCATE
  - DELETES ALL ROWS from a table

Database and some queries based on and/or directly from Querying Microsoft SQL Server 2012 Training Kit by Itzik Ben-Gan, Dejan Sarka, Ron Talmage. ISBN 978-0-7356-6605-4