**Spring 2015 SQL Class**

**Week 2 – Class Notes**

* What problem(s) did we create while doing this weeks assignments that will have to be corrected? What questions do you have?
* [**Database\_Basics\_101**](#Database_Basics_101)
* The example below is a more complete version of the 3 statements we talked about last week, and 3 more that go with it.

SELECT [{ALL | DISTINCT}] select\_item [AS alias] [,…]

FROM [ONLY | OUTER] {table\_name [[AS] alias] | view\_name [[AS] alias]} [,…]

[ [join\_type] JOIN join\_condition ]

[WHERE search\_condition] [ {AND | OR | NOT} search\_condition […] ]

[GROUP BY group\_by\_expression{group\_by\_columns | ROLLUP group\_by\_columns |

CUBE group\_by\_columns | GROUP SETS ( grouping\_set\_list ) |

( ) | grouping\_set , grouping\_set\_list}

[HAVING search\_conditon] ]

[ORDER BY {order\_expression [ASC | DESC]} [,…] ]

* **Substring** is used often you can find more about it at <http://msdn.microsoft.com/en-us/library/ms187748.aspx>
* Dealing with **NULLS** is something that can’t be ignored. The ISNULL() function accepts a variable or expression and tests it for a NULL value. If the value is indeed NULL, then the function returns some other pre-specified value. If the original value is not NULL, then the original value is returned. The syntax is:

ISNULL(<expression to test>, <replacement value if null>)

* **SELECT CLAUSE**

SELECT DISTINCT

SELECT TOP (5)

SELECT TOP (25) PERCENT

* **ORDER BY**

ASC is default

ORDER BY student\_id DESC **,** section

* **WHERE**
  + Each column in a table has an associated datatype which defines what type of data the column can contain. The following link explains the various datatypes that are available in Transact SQL which is the particular dialect of SQL that MS SQL Server uses: <http://msdn.microsoft.com/en-us/library/ms187752.aspx> the most used types are: Numeric, Int, Money, Date, Char, Varchar

**Operator Description**

= Equality

<> Nonequality

!= Nonequality

< Less than

<= Less than or equal to

!< Not less than

> Greater than

>= Greater than or equal to

!> Not greater than

BETWEEN Between two specified values

IS NULL Is a NULL value

LIKE Uses wildcards

EXISTS Returns a simple TRUE/FALSE

* **WILDCARDS** 
  + % represents any length of characters
  + \_ indicates any one character
* **AND** requires that both conditions be met for a row to be retrieved
* The **OR** operator retrieves rows that match either condition, so if one matches and one does not, the row would still be retrieved.
* **NOT** negates whatever statement you bundle with it
* **IN**  Returns TRUE if the value to the left of the IN keyword matches any of the values in the list provided after the IN keyword.
* **Order of Execution**  AND ranks higher in the order of evaluation, whenever you use both AND or OR operators, use parentheses to explicitly group the operators.

* **CAST** and **CONVERT** are used frequently, they are used to perform data type conversions. You can read more about them at <http://msdn.microsoft.com/en-us/library/ms187928.aspx>
* **Scalar Functions** return a single value, based on the input value
  + String Functions <http://msdn.microsoft.com/en-us/library/ms181984.aspx>
  + Numerical Functions <http://msdn.microsoft.com/en-us/library/ms177516.aspx>
  + Date/Time Functions <http://msdn.microsoft.com/en-us/library/ms186724.aspx>
* Read chapters 4 & 5 in our texts to get more information about current discussion topics.
* **Watch your email for Week 2 Homework – the Queries do keep coming!**

**Databases for Mere Mortals**

The concept of **Relational Databases** started in the late 60s and was originally implemented in the early 70s by DR E F Codd an IBM Research Scientist.

**SQL** is the standard language used to create, modify, maintain, and query relational databases.

A **database** is a collection of objects which includes all the tables, views, stored procedures, functions, and other objects necessary to build a database system. Our goal is to develop databases that model some real world system.

There are both **SYSTEM** and **USER** **Objects**. System objects are created during the installation process and are required by the server to run properly. Our mission is to learn how to work with user objects.

**Data** is stored in **RELATIONS**, which are perceived by the user as **tables**. **Tables** contain information about a subject and must have a unique name.

* The physical order of the records or fields in a table is completely immaterial
* Each table must contain a **unique identifier**, a **primary key** which is a single column or a set of columns that uniquely identifies each row in a table.
  + If no appropriate ‘natural key’ can be found a **derived key** or artificial key will need to be created.
* These characteristics allow the data to exist independent of the way it is physically stored in the computer.
* Each table must store data about a single entity or subject, which represents a real life object or event.
  + When a table represents an object, it represents something that is tangible, such as a person, place, or thing. Vendors, customers, products, patients, materials, components, properties, and locations.
  + When the subject of a table is an event, it represents something that occurs at a specific point in time. Appointments, transactions, inquiries, sales, visits, and transfers.
* There are two basic types of tables: Data tables and validation tables.
  + A table that stores data used to supply information is called a data table Data in this type of table is dynamic because it is manipulated (modified, deleted, and so forth) and processed into information in some form or fashion Lots of User interaction.
  + When a table holds data specifically used to implement data integrity, it is known as a validation table. A validation table usually represents subjects such as city names, skill categories, product codes, and project identification numbers. Data in this type of table is static because it will very rarely change at all. No direct user interaction.
* Each **relation or table** is composed of **TUPLES**, or **Records**, or **Rows,**  and **ATTRIBUTES** or **Fields** or **Columns**
  + Each **field** has a unique name and contains a specific piece of information or characteristic about the row or subject of the table in which the field resides
  + Each **row** contains a unique individual occurrence of the subject modeled by the table.

**Relationships** exist between pairs of tables and is established implicitly through matching values of a shared field.

* The Primary Key.
* The Secondary or **foreign key** in a table is the field that relates to the primary key in a second table.
* Relationships are categorized as one-to-one, one-to-many, and many-to-many.