SQL Querying Syntax for Specs (quickie primer)

Two major categories of databases, Relational and “NoSQL”

**Relational** – (~1970). This model organizes data into one or more tables (or "relations") of rows and columns, with a unique key for each row. Generally, each entity type described in a database has its own table, the rows representing instances of that type of entity and the columns representing values attributed to that instance.

            “**NoSQL” –** (~2009) Next Generation Databases mostly addressing some of the points: being non-relational, distributed, open-source and horizontally scalable. Often more characteristics apply such as: schema-free, easy replication support, simple API, eventually consistent / BASE (not ACID), a huge amount of data and more.

**SQL –** Structured Query Language

**SQL Server and DB2 (AES data warehouse) are relational databases. So in order to promote clarity and understanding, it’s beneficial if specs use key words in a way that matches their SQL meaning.**

AND

/\*

       less/fewer results than OR

       all conditions must be true for a single record

\*/

       FirstName = 'Joe'

       AND

       FirstName = 'Jane'

-- no results returned from this query a record’s FirstName value cannot equal both 'Joe' and 'Jane' at the same time.

OR

/\*

       more results than AND

       any of the conditions must be true for a record

\*/

       FirstName = 'Joe'

OR

       FirstName = 'Jane'

IN and NOT IN

/\*

       preferred to multiple "equal to" evaluation

\*/

       FirstName IN ('Joe', 'Jane')

       FirstName NOT IN ('Joe', 'Jane')

BETWEEN

/\*

       inclusive

\*/

       Birthdate BETWEEN '1/1/1990' AND '1/1/2000'

       AND

       DaysDelinquent BETWEEN 1 AND 5 – BETWEEN is inclusive meaning both values of 1 and 5 are included

LIKE and NOT LIKE

/\*

       used for starts with, ends with, and contains

% means anything or nothing

\*/

       RequestName like 'FED%' --starts with FED

       RequestName like '%FED' --ends with FED

       RequestName like '%FED%' --contains FED

NULL

/\*

       NULL means unknown or undefined and is not the same as blank

       checking equal to would not make sense, as cannot determine true/false for exactly equals to something that is unknown

\*/

       MiddleName IS NULL -- is unknown (not recorded)

       MiddleName = '' -- is known to not exist

       MiddleName IS NOT NULL -- is known to exist and

       Middlename = NULL --invalid; can't determine true/false

numbers

/\*

       numbers don't need quotes (quotes cause numbers to be converted to strings and then compared)

\*/

       AccountBalance < 50.00

       ActionCode = '05' -- leading zero requires use of quotes

ordering and dates

ORDER BY

/\*

       ordering is based on the data type the value is stored as (e.g. date vs string (of characters))

       values are ordered either alphabetically (strings and numbers) or chronologically (dates);

if a date is stored in the database as a string (CDW, UDW, ODW do this in many places ☹) it will be ordered alphabetically

\*/

nesting and grouping

/\*

use parenthesis to group (required when OR’s are involved)

the convention is to use tabbing to show nesting

\*/

WHERE

(

Position like 'Manager of%'

)

OR

(

Age > 75

AND

(

HireDate <= '1/20/2008'

OR

TerminationDate > '1/1/2015'

)

)

**Next Level:**

* Calling out possibilities of NULL values (due to the need for special handling when ordering or doing calculations (e.g. 456 + NULL = ERROR; NULL values order before all other values)
* Planning for possibility of future changes
  + For example, if a new type was added, would we most likely want it included or excluded in the query results?
    - Use IN or NOT IN to accommodate most likely future changes