Lesson Material

Were there any sales in the database completed at an inactive store?
Which sales included tequila products?
Which distinct products were sold in Mason City, IA?
Which Scotch whiskies were sold in Mason City, IA?
Which unique types of products, other than whiskies, were sold in Mason City, IA?
List all products and the stores IDs they were sold at (or use the default of ‘No Sale’ if the product was not sold).
As a check for data consistency, were there any sales of products that are not listed in the product table?
As another check for data consistency, were there any sales occurring at a store that does not exist?

Projects

Case Statements and Sub selects

I never guess. It is a capital mistake to theorize before one has data. Insensibly one begins to twist facts to suit theories,

instead of theories to suit facts.

-Sir Arthur Conan Doyle, Author

- Overview
- Case Statements
- Nulls and Zeros
- Sub Selects

Case Statements

```
SELECT field1, field2,
CASE
WHEN field1 >= A THEN ‘field or expression'
END AS New_field_name
FROM  Table1
```

Case Statements

```
SELECT field1, field2,
CASE
WHEN field1 >= A THEN ‘field or expression'
WHEN field1 >= B AND field1 < C THEN ‘field or expression'
WHEN field1 < CTHEN ‘field or expression'
END AS New_field_name
FROM  Table1
```

COALESCE(field1, field2, ‘value’)

SQL

COLESCE – returns the first field that is not null

COLOR	NUMBER	STATE
Blue	1	CA
NULL	300	NULL
NULL	NULL	WA
Yellow	42	NULL

COLESCE (STATE, COLOR, NUMBER, 'UNKNOWN')

COLESCE	COLOR	NUMBER	STATE
CA	Blue	1	CA
300	NULL	300	NULL
WA	NULL	NULL	WA
Yellow	Yellow	42	NULL

Adding Logic to Query

Case Statements

For the purposes of this example, say large is equal to or greater than 400,000, medium is 100,000 and less then or equal to 400,000 and small is anything lower than 100,000.

```
SELECT county, population,
CASE
WHEN population >= 400000 THEN 'large'
WHEN population >= 100000 AND population < 400000 THEN 'medium'
WHEN population < 100000 THEN 'small'
END AS county_size
FROM  counties
```

Case Statements

Now lets say we wanted to get averages on all whiskies regardless of type:

```
SELECT AVG(CASE
WHEN category_name LIKE '%WHISKIES%'
THEN 1
ELSE 0
END) as AverageWhisky
FROM products
```

Sub select

Categories Items by Btl_price as:

High >500

Medium Between 500 and 100

Low <100

```
btl_price - state_btl_cost < 25 THEN 'Low_profit'
btl_price - state_btl_cost BETWEEN 25 AND 250
THEN 'Medium_profit'
ELSE 'High_Profit'
END ‘profit’
```

Commenting

"Always write your code as though a psychopath that knows where you live will be supporting it."

John F Woods
CFO Mitsubishi Financial Group

Sub Selects

```
SELECT field2, AVG(feild3)
FROM
(
        SELECT field1, feild2, count(*) as  field3
        FROM table1
        WHERE field3 >X
        GROUP BY  field1, field2) as T1


GROUP BY T1.field2
```

Sub Selects

1.Which counties have the highest sale of whisky or vodka products?

2.Which counties have the highest sale of items over 80 proof?

3.What are the top 5 counties in terms of mL per capita? How do they compare to the counties you recommended for the pilot program?

4. What percentage of sales per county are over $100? What are the top 5 counties?

5.What were the top 5 categories of liquor sold (based on number of sales) in the five most populous counties?

String Math Date functions

Applying Functions in SQL

String Math Date functions

Applying Functions in SQL

String Math Date functions

The price of light is less

than the cost of darkness.

Arthur C. Nielsen, Founder ACNielsen Company

String Math Date functions

Apply string functions to manipulate how data is presented.

Apply math functions to add value to the data you are working with.

Apply date logic to your SQL.

Review and Overview

- What is a subselect?
- When would you use a subselect?
- What does a subselect syntax look like?

String Math Date functions

- String Functions
- Numeric Functions
- Time and Date Functions
- Set Functions
- Distinct Set Functions
- Join Operators
- Predicate Operators
- Expression Operators
- Boolean Operators
- Data Type Conversion Operators
- Value Expressions

Operators

Operator type	Character(s)	Description
Arithmetic	+ - * / **	Addition or prefix plus Subtraction or prefix minus Multiplication Division Exponentiation
Comparison	= ¬= < ¬< > ¬> <= >=	Equal to Not equal to Less than Not less than Greater than Not greater than Less than or equal to Greater than or equal to
Logical	¬ & |	Not, Exclusive-or And Or
String	||	Concatenation

Delimeters

Name	Delimiter	Use
Comma	,	Separates elements of a list; precedes the BY NAME option
Period	.	Connects elements of a qualified name; decimal or binary point
Semicolon	;	Terminates a statement
Equal sign	=	Indicates assignment or, in a conditional expression, equality
Colon	:	Connects prefixes to statements; connects lower-bound to upper-bound in a dimension attribute; used in RANGE specification of DEFAULT statement
Blank	b	Separates elements
Parentheses	( )	Enclose lists, expressions, iteration factors, and repetition factors; enclose information associated with various keywords
Locator	-> =>	Denotes locator qualification (pointers and offsets) Denotes locator qualification (handles)
Percent	%	Indicates %statements and %directives
Note: Omitting certain symbols can cause errors that are difficult to trace. Common errors are unbalanced quotes, unmatched parentheses, unmatched comment delimiters, and missing semicolons.		

Expressions and Arguments.

- Expression operators create an expression, used to change or modify values returned.
  - Expressions in SQL generally fall into one of four categories including: Boolean, Numeric, Character, and/or Date Expressions.
- An argument can be a literal value, a variable, or an expression in a SQL statement.

String Math Date functions

- CONCAT: Combines two fields or expressions together.
  - SYNTAX
    - CONCAT(field1, field2, field3…)
  - EXAMPLE
    - SELECT CONCAT(item,' - ',description)
    - FROM sales
- LIMIT 100

Learning Objectives

- LOWER: Converts a field or expression to lowercase
  - SYNTAX
    - LOWER(field1)
  - EXAMPLE
    - SELECT LOWER(CONCAT(item,' - ',description))
    - FROM sales
- LIMIT 100

String Math Date functions

- UPPER: Converts a field or expression to uppercase.
  - SYNTAX
    - UPPER(field1)
  - EXAMPLE
    - SELECT UPPER(LOWER(CONCAT(item,' - ',description)))
    - FROM sales
- LIMIT 100

String Math Date functions

- LEFT: Trims number of characters from left side as indicated in the length portion of syntax.
  - SYNTAX
    - LEFT(field1, length)
  - EXAMPLE
    - SELECT CONCAT(item,' - ',description), LEFT(UPPER(LOWER(CONCAT(item,' - ',description))),5)
    - FROM sales
    - LIMIT 100

String Math Date functions

- RIGHT: Trims number of characters from left side as indicated in the length portion of syntax.
  - SYNTAX
    - RIGHT(field1, length)
  - EXAMPLE
    - SELECT CONCAT(item,' - ',description), RIGHT(UPPER(LOWER(CONCAT(item,' - ',description))),5)
    - FROM sales
    - LIMIT 100

String Math Date functions

- LTRIM: Trims all blanks from the left side.
  - SYNTAX
    - LTRIM(field1)
  - EXAMPLE
    - SELECT CONCAT(item,' - ',description), LTRIM(RIGHT(UPPER(LOWER(CONCAT(item,' - ',description))),5))
    - FROM sales
    - LIMIT 100

String Math Date functions

- RTRIM: Trims all blanks for the right side.
  - SYNTAX
    - RTRIM(field1)
  - EXAMPLE
    - SELECT CONCAT(item,' - ',description), RTRIM(RIGHT(UPPER(LOWER(CONCAT(item,' - ',description))),5))
    - FROM sales
    - LIMIT 100

String Math Date functions

- LENGTH: Counts the length of characters in a field.
  - SYNTAX
    - LENGTH(field1)
  - EXAMPLE
    - SELECT LENGTH(CONCAT(item,' - ',description))
    - FROM sales
- LIMIT 100

String Math Date functions

•REPLACE: Similar to the Excel function substitute. It allows you to replace a value in a field with another value.

- SYNTAX
  - REPLACE(field1, field2, field3)
    - Field1 is the field you want to change
    - Field2 is what you want to change in field1
    - Field3 is what you want to change the value in Field2 into
- EXAMPLE
  - SELECT REPLACE((CONCAT(item,' - ',description)),'Absolut', 'Grey Goose')
  - FROM sales
  - LIMIT 100

String Math Date functions

•SUBSTRING this function allows you to isolate a section of characters within a field to retrieve.

- SYNTAX
  - SUBSTRING(field1, start, length)
  - SUBSTRING(field1, starting position, number of characters to retrieve from starting positions)
- EXAMPLE
  - SELECT SUBSTRING(REPLACE((CONCAT(item,' - ',description)),'Absolut', 'Grey Goose'),9,35)
  - FROM sales
  - LIMIT 100

String Math Date functions

•TRIM: Removes characters from start of field, ending part of field or both that are indicated in the formula.

- SYNTAX
  - TRIM(leading ‘characters’, from field1)
  - TRIM(trailing ‘characters’, from field1)
  - TRIM(both ‘characters’, from field1)
- EXAMPLE
  - SELECT description, TRIM(Leading 'A' from description), TRIM (TRAILING 'a' from description), TRIM(BOTH 'A' FROM description)
  - FROM sales

•LIMIT 100

String Math Date functions

•ROUND: This function returns the specified number rounded to the specified integer; the number rounded to the specified places to the right of the decimal point. If the integer is negative, the number is rounded to the specified places to the left of the decimal point.

- SYNTAX
  - ROUND(numeric_expression, integer_expresssion)
- EXAMPLE
  - ROUND(177.3589,2) would return 177.3600
  - ROUND(177.3589,-2) would return 200

String Math Date functions

- CURRRENT_ DATE brings back the current date from the system.
  - SYNTAX
    - CURRENT_DATE
  - EXAMPLE
    - SELECT item, total, date, CURRENT_DATE
    - FROM sales
    - LIMIT 100

String Math Date functions

- AGE this function brings back the difference between two dates.
  - SYNTAX
    - Age(date1, date2)
  - EXAMPLE
    - SELECT item, total, date, current_date, age(date, current_date)
    - FROM sales
    - LIMIT 100

String Math Date functions

- **INDEPENDENT PRACTICE: DATE FUNCTIONS** 15 mins
- CALCULATE AGE
- Find the age of all the stores and summarize their sales.

CREATE SPECIAL DAYS

- Using Dates and basic math Make 1 Select statement that holds a field for each one of these Special days
- TODAY
- TODAY LAST YEAR
- YESTERDAY
- YESTERDAY LAST YEAR
- LAST WEEK
- LAST WEEK LAST YEAR

String Math Date functions

- **INDEPENDENT PRACTICE: DATE FUNCTIONS** 15 mins
- CALCULATE AGE
- Find the age of all the stores and summarize their sales.

CREATE SPECIAL DAYS

- Using Dates and basic math Make 1 Select statement that holds a field for each one of these Special days
- TODAY
- TODAY LAST YEAR
- YESTERDAY
- YESTERDAY LAST YEAR
- LAST WEEK
- LAST WEEK LAST YEAR

String Math Date functions

- **INDEPENDENT PRACTICE: DATE FUNCTIONS** 15 mins
- CALCULATE AGE
- Find the age of all the stores and summarize their sales.

CREATE SPECIAL DAYS

- Using Dates and basic math Make 1 Select statement that holds a field for each one of these Special days
- TODAY
- TODAY LAST YEAR
- YESTERDAY
- YESTERDAY LAST YEAR
- LAST WEEK
- LAST WEEK LAST YEAR

String Math Date functions

- **INDEPENDENT PRACTICE: DATE FUNCTIONS** 15 mins
- CALCULATE AGE
- Find the age of all the stores and summarize their sales.

CREATE SPECIAL DAYS

- Using Dates and basic math Make 1 Select statement that holds a field for each one of these Special days
- TODAY
- TODAY LAST YEAR
- YESTERDAY
- YESTERDAY LAST YEAR
- LAST WEEK
- LAST WEEK LAST YEAR

String Math Date functions

- **INDEPENDENT PRACTICE: DATE FUNCTIONS** 15 mins
- CALCULATE AGE
- Find the age of all the stores and summarize their sales.

CREATE SPECIAL DAYS

- Using Dates and basic math Make 1 Select statement that holds a field for each one of these Special days
- TODAY
- TODAY LAST YEAR
- YESTERDAY
- YESTERDAY LAST YEAR
- LAST WEEK
- LAST WEEK LAST YEAR

String Math Date functions

CALCULATE AGE

•Find the age of all the stores and summarize their sales.

CREATE SPECIAL DAYS

•Using Dates and basic math Make 1 Select statement that holds a field for each one of these Special days

•TODAY

•TODAY LAST YEAR

•YESTERDAY

•YESTERDAY LAST YEAR

•LAST WEEK

•LAST WEEK LAST YEAR

String Math Date functions

Categorize all of the items based on list date and ranges of 0-10 years, 11-20 year, 21-30 years, 31-40 years and 41+ years, then bring in the total sales.

USING and a LIKE trick

String Math Date functions

```
DROP TABLE IF EXISTS NEWTABLE;
CREATE TABLE NEWTABLE as
(
SELECT FIELD1, FIELD2, FIELD3
FROM "TABLE1"
);
```

String Math Date functions

Applying Functions in SQL