SQL Unit 4
Built-in Functions and Calculated Fields

Ryan Nixon

Reminder From Wednesday

- Please review the schedule changes on the website
- We are moving beyond Microsoft Access. SQLite is a recommended alternative

Functions

- Predefined actions applied to fields within a SELECT statement
- Manipulates information as it is retrieved without modifying the original data
- May be aggregate, row-level, or calculated
- “Returns” information (a term from programming)
- SELECT FUNCTION(column) AS alias

Taking Notes?

- I'll provide a printable function reference on the website so that you know the purpose of each function
- This reference will not be allowed on the test
- It is still up to you to memorize and learn to use the functions

Aggregate Functions

- Defn: "A whole formed by combining several (typically disparate) elements."
- Calculates results from the combination of all rows
- Cannot apply to individual rows or groups of rows
- Cannot be mixed with other non-Aggregate functions

COUNT

- Returns the number of non-Null fields in a resultset
- Can apply to 1 field or all of them (i.e. *)
- Applied to the primary key, returns the total number of rows in the resultset
- Will not count Nulls in the field it is pointing to

COUNT

-
  ```
  SELECT COUNT(id)
  FROM person
  WHERE first LIKE ‘A%’
  ```
  Returns 36

-
  ```
  SELECT COUNT(id)
  FROM person
  ```
  Returns 521

Introducing...
Your Test Scores

MAX, MIN, SUM, AVG...

- MAX - Find the maximum value of a field in a table
- MIN - Find the minimum value of a field in a table
- SUM - Find the sum of the values of a field in a table
- AVG - Find the average of the values of a field in a table
- Many many more, but your compatibility may vary.
- Like COUNT, these functions do not look at Null as a value

Notice the Single Row

- Aggregates only return a single result. This is because they calculate off of the whole resultset and throw all of the row data away.
- This makes these functions unable to mix with the typical query results we've seen so far
- Next week we'll see how to partially mix results with aggregates

Row Functions

- Unlike aggregates, these apply to each row
- Can be mixed in with the typical query
- Modify the expected result in some defined way

Mathematical Functions

- ABS - Find the absolute value of a field in a table.
- ROUND(X,Y) - Round off a numeric value to a specified number of decimal places.
- Note that ROUND takes two parameters
    - The column to be rounded
    - The number of decimal places to round to. Default is 0

Mathematical Functions

- RANDOM (SQLite) / RAND (Others)
- Returns a random number between 0 & 1
- Why is this useful?
  - Combine with ROUND to assign random integers to rows
  - Put in the ORDER BY to randomize order of results

String Functions

- UPPER, LOWER - Convert text to upper or lower case, respectively
- TRIM, LTRIM, RTRIM - Remove white-space off of the ends of a string or just the left/ right sides

String Functions

- `SELECT UPPER(first) AS name`
  `FROM person`
  Returns “   JAMIE   ”
- `SELECT LTRIM(LOWER(first)) AS name`
  `FROM person`
  Returns “jamie   ”

“Calculated” Fields

- Fancy term for “straight arithmetic works too”
- Mathematical operators such as + - * / and parentheses all work in SELECT statements
- Act just like row-level functions, applying to each individual row
- Aliases are very important here

String Concatenation

- An odd exception
- Uses a mathematical operator `||` in SQLite, but the `CONCAT` function in other languages
- Both can take any number of parameters
- NOTE - Don’t mix these! `||` has an entirely different meaning outside of SQLite

String Functions

- `SELECT first || ‘ ‘ || last AS name`

  `FROM person`

  SQLite - Returns “Marie Woods”

- `SELECT CONCAT(first, ‘ ‘, last) AS name`

  `FROM person`

  Other SQL - Returns “Marie Woods”

Date/Time Functions

- A collection of row-level functions that manipulate dates and/or times
- Many data types to work with; we'll focus on these three:
  - Date - Contains only date information ('2012-10-01')
  - Time - Contains only time information ('16:40:00')
  - Datetime - Contains both sets of information ('2012-10-01 16:40:00')

Date/Time Functions

- Sometimes when dealing with dates, a "cast" must be made. This forcibly converts data from one type to another, possibly losing information in the conversion
- When using strings with dates in them, a cast may be required before SQL will consider it a date

Date/Time Functions

- One more bit of bad news.
- Date/Time functions vary wildly between SQL implementations, so I will only be able to cover a small amount

Casting Functions (MySQL, MS SQL)

- DATE - 2012-10-01
- TIME - 16:43:00
- YEAR - 2012
- MONTH - 10
- DAY - 1

Casting Functions (MySQL, MS SQL)

- `SELECT YEAR(dob)`
  `FROM person`
  Returns ‘1979’
- `SELECT MONTH(dob)`
  `FROM person`
  Returns ’10’

Casting Functions (SQLite, MySQL)

- STRFTIME...huh?
- This function is used for formatting, so `STRFTIME(‘%Y’, dob)` would produce ‘1979’
- Much more powerful & flexible, but much more confusing to learn
- MySQL also supports this: `DATE_FORMAT(dob, ‘%Y’)`

Casting Functions (SQLite, MySQL)

- %Y - year (0000-9999)
- %m - month (01-12)
- %d - day (01-31)
- %H - hour (00-23)
- %M - minute (00-59)
- %S - seconds (00-59)
- %j - day of year (001-366)
- %w - day of week (0-6)
- %% - %

Casting Functions (SQLite)

- `SELECT STRFTIME(‘%Y’, dob)`
  `FROM person`
  Returns ‘1979’

- `SELECT STRFTIME(‘%m’, dob)`
  `FROM person`
  Returns ’10’

What about now?

- You may use fields from the database in your functions, but what about the current datetime?
    - GETDATE (MS SQL)
    - NOW (MySQL)
    - DATETIME() or ‘now’ in STRFTIME (SQLite)
- Returns a fully formatted date that can be cast by the aforementioned functions

Example: 2012-10-01

- MS SQL:

```
SELECT DATE(GETDATE()) AS date
```

- MySQL:

```
SELECT DATE(NOW()) AS date
```

- SQLite:

```
SELECT STRFTIME(‘%Y-%m-%d’,
DATETIME()) as date
```

Example -17:05:00

- MS SQL:
  ```
  SELECT TIME(GETDATE()) AS time
  ```
- MySQL:
  ```
  SELECT TIME(NOW()) AS time
  ```
- SQLite:
  ```
  SELECT STRFTIME(‘%H:%M:%S’, ‘now’)
  as time
  ```

Limiting Results

- TOP (MS SQL), LIMIT (Other SQL)
- Apply to whole resultset and all columns
- Limit the number that are returned
- Number is placed after declaration, *not* in parentheses
- Locations are also different per implementation

Limiting Results

- Microsoft SQL Server:

```
SELECT TOP 10 *
FROM person
```

- MySQL / SQLite:

```
SELECT *
FROM person
LIMIT 10
```

Reminders

- Assignment 4 up tonight. Due 10/22
- Lab time Wednesday in SSB 172
- Test & assignment grades are on Blackboard