Multi-Paradigm Programming - Declarative Programming

Dominic Carr

Galway-Mayo Institute of Technology

dominic.carr@gmit.ie

What We Will Cover

Goals of this Session

Goals

- To understand....
  - What is Declarative programming?
  - How is it different from what we have seen?
  - Why is valuable?

Declarative Programming

Declarative I

Programming by specifying the **result you want**, not *how* to get it.

```sql
select upper(name)
from people
where length(name) > 5
order by name
```

- Control flow is implicit: the programmer states only what the result should look like, not how to obtain it.
- No loops, no assignments, etc.
- Whatever engine that interprets this code is supposed go get the desired information, and can use whatever approach it wants.
- SQL is declarative

Declarative II

```
Select * from Worker order by FIRST_NAME asc;
```

- We would naturally assume the results will be pulled from a database table, but really the declaration of what we want is independent of how it it retreived
  - Could be from a text file, could be a human has to type out the response
  - The data store could be local or it could be on an AWS host in another country
  - The point of the declarative lanaguage is to abstract somewhat from the how.

Declarative III

- Declarative programming
    - contrasts with imperative and procedural programming.
    - Declarative programming is a *non-imperative style of programming*
        - Describe result not being explicit about how to obtain them.

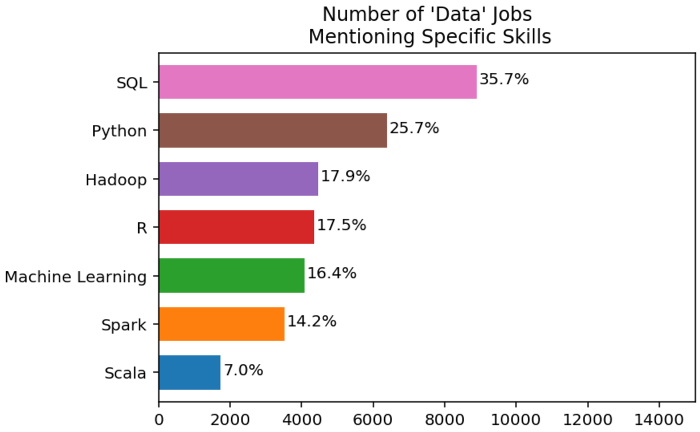
SQL

- Structured Query Language
- Domain-specific language
  - Designed for managing data held in a relational database management system (RDBMS)
  - CRUD
- It is particularly useful in *handling structured data*, i.e. data incorporating **relations** among entities and variables.
- Dates back to 1970
  - Edgar Codd wrote a paper describing a new system for organizing data in databases.
  - By the end 70's, prototypes of Codd's system had been built, and a query language — **the Structured Query Language (SQL)** — was born to interact with these databases.

Popularity of SQL

COMPANIES USING SQL

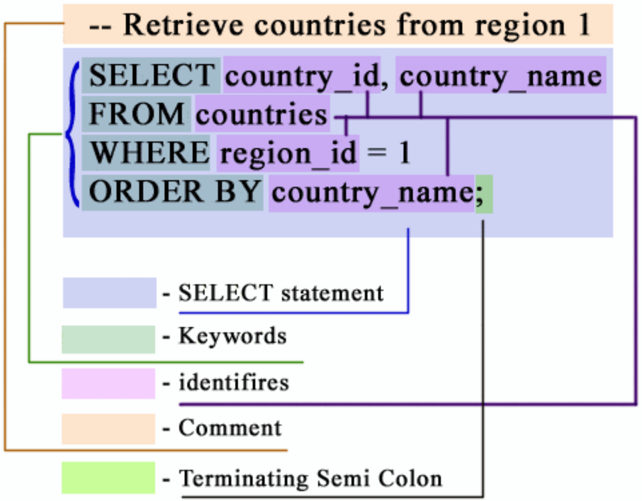
Popularity of SQL



SQL Syntax

- The SQL language is subdivided into several language elements, including:
  - Clauses, which are constituent components of statements and queries. (In some cases, these are optional.)
  - Expressions, which can produce either scalar values, or tables consisting of columns and rows of data
  - Predicates, which specify conditions that can be evaluated to SQL three-valued logic (3VL) (true/false/unknown) or Boolean truth values and are used to *limit the effects of statements and queries*, or to change program flow.
  - Queries, which **retrieve the data** based on *specific criteria*. This is an important element of SQL.
  - Statements, which may have a **persistent effect** on schemata and data, or may control transactions, program flow, connections, sessions, or diagnostics.

SQL Language Elements



```
-- Retrieve countries from region 1
SELECT country_id, country_name
FROM countries
WHERE region_id = 1
ORDER BY country_name;
```

Standard SQL

```sql
SELECT OrderID, Quantity,
CASE WHEN Quantity > 30 THEN "The quantity is greater than 30"
WHEN Quantity = 30 THEN "The quantity is 30"
ELSE "The quantity is under 30"
END AS QuantityText
FROM OrderDetails;
```

Comparison with C

Imagine the schema looks like:

personId INTEGER PRIMARY KEY, name VARCHAR(20), sex CHAR(1), birthday DATE, placeOfBirth VARCHAR(20));

How would we find the oldest person in the table?

```
SELECT * FROM people
 ORDER BY birthday ASC
 LIMIT 1
```

Equivilant C I

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

struct Person {
    char* name;
    char sex;
    int age;
    char* placeOfBirth;
};

int main(void)
{
    struct Person a = { "Anthony Stynes", 'M', 12, "New York" };
    struct Person b = { "Bertie Corr", 'M', 1000, "Boston" };
    struct Person c = { "John Quin", 'M', 33, "'Murica" };
```

Equivilant C II

```
    struct Person arr[] = { a,b,c };
    int maxAge = -1;
    int index = 0;

    for(int i = 0; i< 3; i++){
        if (arr[i].age > maxAge){
            maxAge = arr[i].age;
            index = i;
        }
    }
    printf("%s is oldest\n", arr[index].name);

    return 0;
}
```

What's the most complex SQL query you ever wrote?

Greg Kemnitz, Postgres internals, embedded device db internals, MySQL user-level

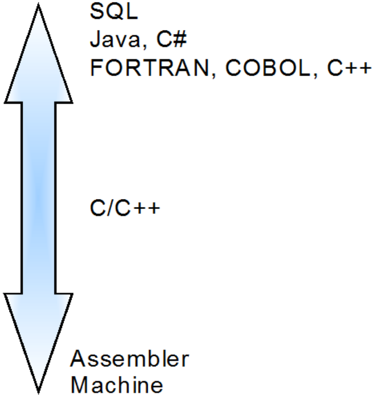
"I wrote a reporting query once that **was 700 lines long** and visited **27 different tables** in lookups or joins. It didn't do any fancy stuff - just straight whereclause predicates and joining - but it was pretty gnarly and **took 3 days to compose, debug, and tune**."

Bill Karwin, author of "SQL Antipatterns: Avoiding the Pitfalls of Database Programming"

"One of the fun, complex SQL queries I wrote was for a demo I did during a talk at OSCON 2006, SQL Outer Joins for Fun and Profit. **It was a query that solved Sudoku puzzles**."

Level of Abstraction

- High-level
  - Close to problem
  - System independent
- Low-level
  - Close to system
  - Doesn't reflect problem



Make

- Make is a build automation tool
  - automatically builds executable programs and libraries from source code
  - ....by reading files called Makefiles which specify how to derive the target program.
- Besides building programs, Make can be used to manage any project where some files must be updated automatically from others whenever the others change.

```
CFLAGS ?= -g

all: helloworld

helloworld: helloworld.o
    # Commands start with TAB not spaces
    $(CC) $(LDFLAGS) -o $@ $^

helloworld.o: helloworld.c
    $(CC) $(CFLAGS) -c -o $@ $<

clean: FRC
    rm -f helloworld helloworld.o

# This pseudo target causes all targets that depend on FRC
# to be remade even in case a file with the name of the target
    exists.
# This works with any make implementation under the assumption
    that
# there is no file FRC in the current directory.
FRC:
```

Sources

Sources

- https://www.computerhope.com/jargon/i/imp-programming.htm
- https://en.wikipedia.org/wiki/Abstraction_(computer_science)
- https://www.quora.com/Whats-the-most-complex-SQL-query-you-ever-wrote
- https://en.wikipedia.org/wiki/Make_(software)#Makefile

The End