

OT: SQL

Source: <http://sci.tech-archive.net/Archive/sci.logic/2008-05/msg00124.html>

- *From:* Marshall <marshall.spight@xxxxxxxxxx>
 - *Date:* Sat, 3 May 2008 16:01:36 -0700 (PDT)
-

On May 3, 2:24 pm, Charlie-Boo <shymath...@xxxxxxxxxx> wrote:

On May 3, 12:26 pm, Marshall <marshall.spi...@xxxxxxxxxx> wrote:

Codd noticed that databases contain relations. The rest of what he said was a totally failed attempt to formalize and automate database query processing. SQL is an example of that failure, by virtue of it being only a programming language rather than an automatic query processor.

"Totally failed" is too strong. Partly failed, partly succeeded. Still a definite step forward if you compare it to what came before.

Is there a procedure for asking "List all employees who earn more than their managers." (classic problem)

Um, yeah, that's pretty easy actually.

— first, the data definition:

```
create table Employees
(
  EmployeeId int primary key,
  Salary Money
);
```

```
create table Management
(
  ManagerId int foreign key references Employees(EmployeeId),
  EmployeeId int foreign key references Employees(EmployeeId),
```

```
primary key(ManagerId, EmployeeId)
);
```

-- then the actual query

-- list the employee id of all employees who have a
 -- larger salary than their manager.

```
SELECT e.EmployeeId
FROM Employees e, Employees m, Management mgt
WHERE
mgt.ManagerId = m.EmployeeId AND
mgt.EmployeeId = e.EmployeeId AND
e.Salary > m.Salary;
```

Notice that this solution specifies nothing whatsoever about how the result is to be obtained. No execution plan is mandated. This is a straightforward, unadorned declarative specification of what information is desired.

(If there are any database geeks out there, note also that my solution makes no use of NULL; it stays firmly in 2VL.)

and it figures out different ways
 to do it (then one is chosen somehow)?

Just so. The query planner will consider a number of different query plans and choose the optimum, or the heuristically–best if the search space for best is too large.

In higher–end DBMS products, the query planner will consider different execution types: merge join vs. nested loops vs. whatever. Even in something less sophisticated like MySQL, which only supports nested loop join, the query planner will take into account the existence of different indexes, and choose an order to traverse the tables on that basis. And in fact, the choice may well change based on what indexes exist, which can change *without requiring any change to the above query.* The query has an associated logical semantics, but does not specify an operational semantics. This is called "physical independence," and it's one of the reasons I say SQL is a partial success.

If you want to say SQL has problems, I won't disagree.

In fact I'll probably enthusiastically join in. But it's not a total failure; it gets a number of things right.

Marshall

.