Overall, I not only enjoyed the course, but I also learned a lot from it.

* For instance, being relatively new at using q, it was very helpful for me to explore some of the basic built-in q functions and see examples of how they can be used.
* Also, the problem driven approach to teach this course was helpful since I could immediately see how the concepts we learned could be applied. We would often go through a step-by-step explanation of the solution code, which helped a lot, especially with advanced concepts like tables and inter process communication.
* However, not being familiar with finance and some of the math related concepts in some of the examples, I struggled trying to understand the problem rather than trying to apply to q concepts.

Pros:

* Interactive approach (teach a little, do a little) worked well
* First day especially was very good
* Dennis was engaging, knowledgeable and proactive
* Liked the fact that he was willing to take the risks that come with teaching on-the-fly
* Problems were generally good, fit with the materials, worked out well, taught the right things; finding good problem is hard work
* Having the materials in advance was a definite plus

Areas for improvement:

* Could have done with a few slides to introduce sections, set things up, provide overview and context
* Slides with background for some of the problems (moving average, k-means, etc), etc. would have been helpful since not everyone was familiar with these concepts
* The coverage of tables and q-sql lacked context – felt like it jumped around – it would have benefited from a more systematic walk-through
* The client/server sections were good but I would have liked more depth there; once again, a systematic approach that started with simple cases and then elaborated would be preferred

-Overall, i enjoyed the course, and definitely learned new things that should be valuable to me in my q programming.

-I got a lot out of the discussion of different q functions and how they can be used. I found it useful to get a general breakdown of some handy q functions and where and how they can be applied.

-Being relatively new to both q and databases, I found that the material on tables (foreign keys, special joins etc..) went a little fast. I felt I was re-using the code used in the lesson a bit too much without really understanding as much as I'd have liked to.

-In general, the examples used were mainly dealing with financial formulas and concepts. Not being very familiar with these concepts, I found I was spending a bit too much time trying to understand how the financial examples worked, and less time learning how q works.

-I would have been interested in going into some more detail about the differences in performance between q functions used for certain tasks.

In general I liked the problem driven approach, but..

- In some cases not enough introductory material was provided

 - In others his solution included functions that had not been covered

Although decipherable the use of problems from the financial domain resulted in some impedance.

He tended to refer to verbs by their character symbol rather than their name. For example hash rather than take.

IPC - only covered sending strings, sending lists offers more flexibility. (Ship your own function over etc..)

Tables felt weak  
 - no explanation of why exec was returning a dictionary in one example, need to explain it can also return a list (exec returns a list when you ask for a single column dictionary for more than one)  
 - asked to time tbl vs column insert but offered no insight as to which should be better or why.  
 - you CAN create keyed tables without unique keys, behavior is undefined but it can be done,  q will not error. In particular ([a: 1 1 1]b:2 2 2) will work

 - Explanation/handling of compound column types in queries could have been better explained. An example using a no atomic function in a where would have been useful

 - where: would be nice to show that q-sql where is the same as q where ie, everything after the where is generating a bit vector, if you knew the answer you could just say select from t where 10001b. In my experience this reduces where clauses to something easily understandable, also makes is easy to debug them outside of q-sql .

 - Keyed tables weren't discussed much.

-  Functional forms were not discussed. This is the correct way to build dynamic queries (as opposed to building a string and using value)

Most of these form a wishlist of things I would have liked covered. Of course in a 3 day course there is only so much you can do

Generally I liked the course:

* The delivery of problem solutions enabled me to do useful learning when I became "stuck" on solving a problem
* The learning sequence was appropriate
* The instructor and class dynamics were excellent. I thought the class was a very productive use of my time.

However:

* The course title "...for smarties" did not inform me whether I was expectecd to be a programmer proficient in other languages, or to have knowledge of the subject matter referenced in the problems (e.g. financial), or already have some q experience.
* The solutions to to some problems required some q knowledge which was not presented in advance. It could have been useful if a summary of the prerequisite q knowledge reqwuirements were distributed to students a few days before the course. For future courses, I recommend the leader rigorously review the required knowledge for problem solution.