Practical SQL: A Beginner's Guide to Storytelling with Data

Feedback on the 1st Edition

General comments

- When the joins are mentioned for the first time, the book uses aliases for tables being joined by explicitly using the `AS` keyword (e.g. `left_table AS lt`). But then it stops using this keyword later on (e.g. `left_table lt`). Although, it might not be hard for someone to see that this keyword is optional, I think it's still a good practice for beginners to repeatedly see aliasing using `AS` keyword, so that they embody the habit of making their aliasing intent clear using this keyword. Plus, it may avoid confusion for some other users who may be wondering why there is no `AS`, and if this is an error.
- While discussing writing functions for triggers, I think it would be helpful if the book were to clarify that we don't need to specify the table explicitly, because the function will always pick up the table specified in when `CREATE TRIGGER` clause is specified. Users coming from other programming languages will spend some time scratching their heads as to how does the function *know*, for example in Listing 15-22, which table does `NEW` keyword refer to?!

Possible clarifications

- I wish the book emphasized how DBMS, data model, and query language (SQL) are all different things. Only then will it be clear that, for example, *MySQL*, *PostgresSQL*, etc. are RDBMS, and **not** different dialects of SQL developed for respective databases (which is what I walked away thinking having read the book).
- The way derived tables, CTEs, and view are discussed in the book, sometimes it may feel like they can't all be combined in the same workflow. So, maybe an example that contains more than one of these concepts can be produced to drive this point home:

```
CREATE OR REPLACE VIEW ... AS
WITH ... AS (
...
)
```

Wishlist

- The book does not cover some common terms used while talking about relational data model: *attribute*, *tuple*, *domain*, *degree*, *cardinality*
- I wish the book had also covered the following operators:
  - `UNION` (and `UNION ALL`)
  - `ANY` and `ALL`
- While discussing joins, if not explicit treatment, self joins may at least deserve a hat tip.