

4. At the bottom, there are statistics that give information about the memory usage and execution time. Absolute numbers can be different if you run this code, but the difference between the methods used should be about the same:

|  |  |  |
| --- | --- | --- |
| Method | Memory usage (megabytes) | Execution time (seconds) |
| Active Record | 21.4 | 2.398 |
| Query Builder | 28.3 | 0.477 |
| SQL (DAO) | 27.6 | 0.481 |

How it works...

The actionAr action method gets model instances using the Active Record approach.

We start with the Actor model generated with Gii to get all the actors, and specify joinwith => 'films' to get the corresponding films using a single query or eager loading through relation, which Gii builds for us from innoDB table foreign keys. We then simply iterate over all the actors and for each actor, over each film. Then, for each item, we print its name.

The actionQuery function uses Query Builder. First, we create a query for the current DB connection with \yii\db\Query. We then add query parts one by one with from, joininner, and leftJoin. These methods escape values, tables, and field names automatically. The all() function of \yii\db\Query returns an array of raw database rows. Each row is also an array, indexed with result field names. We pass the result to renderRows, which renders it.