

Top 10 Reasons to Use EntityFieldQuery (EFQ)

Fredric Mitchell

fmitchell

@fredricmitchell

1. Core

class EntityFieldQuery

bit.ly/d7-efq

1. Core

Finds entities based on entity properties, field values, and generic entity meta data

2. Well-Documented

[**bit.ly/d7-efq**](https://bit.ly/d7-efq)

[**bit.ly/d7-efqdoc**](https://bit.ly/d7-efqdoc)

2. Well -Docu ment ed

```
/**
 * Orders the result set by entity-generic metadata.
 *
 * If called multiple times, the query will order by each specified column in
 * the order this method is called.
 *
 * Note: The "comment" and "taxonomy_term" entity types don't support ordering
 * by bundle. For "taxonomy_term", propertyOrderBy('vid') can be used instead.
 *
 * @param $name
 *   'entity_type', 'bundle', 'revision_id' or 'entity_id'.
 * @param $direction
 *   The direction to sort. Legal values are "ASC" and "DESC".
 *
 * @return EntityFieldQuery
 *   The called object.
 */
public function entityOrderBy($name, $direction = 'ASC') {
  $this->order[] = array(
    'type' => 'entity',
    'specifier' => $name,
    'direction' => $direction,
  );
  return $this;
}
```

3. Simple

```
//start
```

```
$query = new EntityFieldQuery();
```

3. Simple

//finish

\$result = \$query->execute();

4. Consumable

\$query

- >entityCondition('entity_type', 'node')
- >entityCondition('bundle', 'article')
- >propertyCondition('status', 1)
- >propertyOrderBy('created', 'DESC');

4. Consumable

//views query

```
SELECT node.created AS node__created, node.nid AS nid
```

```
FROM {node} node
```

```
WHERE (( (node.status = 1) )AND(( (node.type IN ('article')) )))
```

```
ORDER BY node__created DESC
```

4. Consumable

\$query

- >entityCondition('entity_type', 'node')
- >entityCondition('bundle', 'article')
- >propertyCondition('status', 1)
- >propertyOrderBy('created', 'DESC')
- >fieldCondition('field_us_state', 'value', array('MN'))
- >range(0,10)

4. Consumable

```
SELECT node.created AS node__created, node.nid AS nid
FROM {node} node
INNER JOIN {field_data_field_location_state} f_state ON node.nid = f_state.entity_id AND
(f_state.entity_type = 'node' AND f_state.deleted = 0)
WHERE (( (node.status = 1) )
AND(( (node.type IN ('article'))
AND (f_state.field_location_state_value = 'MN') )))
ORDER BY node__created DESC
LIMIT 10 OFFSET 0
```

5. Object-Oriented

class EntityFieldQuery

Proper methods, implements drupal hooks

5. Object-Oriented

```
public function execute() {  
    // Give a chance to other modules to alter the query.  
    drupal_alter('entity_query', $this);  
    $this->altered = TRUE;  
    //more stuff  
}
```

5. Object-Oriented

modularity

extensibility

Symfony?

6. Extensible

```
class NodeEntityFieldQuery extends EntityFieldQuery {  
    public function __construct() {  
        // now we don't need to define these over and over anymore  
        $this  
        ->entityCondition('entity_type', 'node')  
        ->propertyCondition('status', 1)  
        ->propertyOrderBy('created', 'DESC');  
        ...  
    }  
}
```


7. Alterable

```
hook_field_storage_query()
```

```
field_sql_storage_field_storage_query(EntityFieldQuery $query)
```

```
//derivative of SelectQuery
```

```
$select_query = db_select($tablename, $table_alias);
```

7. Alterable

//get all the SelectQuery/Query goodness

hook_query_TAG_alter()

7. Alterable

//OR

bit.ly/d7-efqor

7. Alterable

\$query

```
->leftJoin('field_data_field_archive', 'a', 'node.nid = a.entity_id');
```

```
if ($tid) {
```

```
  $or = db_or()
```

```
    ->condition('a.field_archive_tid', array($tid), 'NOT IN') ->isNull('a.field_archive_tid');
```

```
$query
```

```
  ->condition($or);
```

```
}
```

8. Exception Handling

//be a good neighbor

throw new EntityFieldQueryException

9. View Modes

```
$nids = array_keys($result['node'])
```

```
$nodes = node_load_multiple($nids)
```

```
node_view_multiple($nodes, 'teaser');
```

9. View Modes

//tim cosgrove

bit.ly/d7-viewmodes

10. Beans

drupal.org/project/bean

bit.ly/d7-efqbeans

10. Beans

```
public function view( ... ) {  
    //do EFQ  
    $query->setCustomCondition($bean->filters['foo']);  
    $result = $query->execute();  
    //cycle through $result  
    $nodes_view_mode = $bean->nodes_view_mode['option'];  
    node_view($node, $nodes_view_mode);  
    ...  
}
```

bit.ly/d7-efq

bit.ly/d7-efq2

bit.ly/d7-efq3

Questions??