# EOQualifierSQLGeneration

**Adopted By:** EOAndQualifier, EOKeyComparisonQualifier, EOKeyValueQualifier, EONotQualifier, EOOrQualifier, EOSQLQualifier

**Declared In:** EOAccess/EOSQLQualifier.h

## Protocol Description

The EOQualifierSQLGeneration protocol declares two methods that are adopted by qualifier classes to qualify fetches from a database. One of the methods, **schemaBasedQualifierWithRootEntity:**, is used to provide a qualifier suitable for evaluation by a database from a qualifier suitable for in-memory evaluation. The other method, **sqlStringForSQLExpression:**, is used by concrete subclasses of EOSQLExpression to generate WHERE clauses for SQL statements.

## Instance Methods

### sqlStringForSQLExpression:

– (NSString *)**sqlStringForSQLExpression:**(EOSQLExpression *)*sqlExpression*

Returns a SQL statement suitable for inclusion in a WHERE clause. Invoked from a concrete subclass of EOSQLExpression while it's preparing a SELECT, UPDATE, or DELETE statement.

**See also:** – **whereClauseString** (EOSQLExpression)

### schemaBasedQualifierWithRootEntity:

– (EOQualifier *)**schemaBasedQualifierWithRootEntity:**(EOEntity *)*entity*

Returns a qualifier suitable for evaluation by a database (as opposed to in-memory evaluation). Invoked by an EODatabaseChannel object before it uses its EOAdaptorChannel to perform a database operation.

Whereas in-memory qualifier evaluation uses pointers to resolve relationships, a database qualifier must use foreign keys. For example, consider the qualifier below that is used to fetch all employees who work in a specified department:

```
Department *dept;    // Assume this exists.
EOQualifier *qualifer;

qualifier = [EOQualifier qualifierWithQualifierFormat:@"department = %@", dept];
```

For an in-memory search, the Framework queries employee objects for their department object and includes an employee in the result list if its department object is equal to **dept**. (See the EOQualifierEvaluation protocol description for more information on in-memory searching.)

For a database search, the Framework needs to qualify the fetch by specifying a foreign key value for **dept**. The Framework sends **qualifier** a **schemaBasedQualifierWithRootEntity:** message that creates and returns a new qualifier. Assume that the entity for employee objects has an attribute named **departmentID** and that the primary key value for **dept** is 459, the resulting qualifier specifies the search conditions as:

```
department.departmentID = 459
```

**See also:** – **selectObjectsWithFetchSpecification:editingContext:** (EODatabaseChannel)