Codelab: Queries and Indexes

Updated: March 20, 2015

This hands-on exercise introduces you to different queries that you can perform in the Datastore. It also illustrates various ways to filter your data and a look at a Datastore Indexes.

Contents

What You Will Learn

You will learn:

- How to query your datastore in multiple ways
- Understand and view the datastore Indexes being created

What You Will Do

You will try out multiple queries in the same source code as the previous hands-on exercise.

Get Set Up

You can either use your own project files from the last hands-on or download the final Project ZIP files available for both Java and Python. The files are given below:

- Java Datastore Basics Project ZIP file
- Python Datastore Basics Project ZIP file

You can also clone the projects directly from GitHub:

```
git clone https://github.com/GoogleCloudPlatformTraining/cp300-gae-datastore-java.git
```

OR

```
git clone https://github.com/GoogleCloudPlatformTraining/cp300-gae-datastore-python.git
```

Create Data

Since we are going to try out various queries, it is important that you have sufficient sample data. This will ensure that you do get data records when you try out filtering the data based on certain criteria.

The queries that we shall be looking at primarily filter data on the following fields:

- Conference City
- Max Attendees

Here is the suggested data to create:

1. Create 2 Conference entities with **city = London** and **maxAttendees = 50**
2. Create 1 Conference entities with **city = London** and **maxAttendees = 60**
3. Create 1 Conference entity with **city = London** and **maxAttendees = 30**
4. Create 2 Conference entities with a **city other than London** and choose any values for **maxAttendees**.
5. Create 1 Conference entity with **city other than London** and **maxAttendees = 50**

Java Instructions

We need to make changes to **index.jsp** file to see the various Queries in action. You will need to place the snippet of code in the area highlighted below:

```
<%

    // Get a handle to the datastore service
    DatastoreService datastore =
                   DatastoreServiceFactory.getDatastoreService();

    // Create the Query to get all the conferences
    Query confQuery = new Query("Conference");
    confQuery.addSort("title", SortDirection.ASCENDING);

    <<PASTE CODE HERE>>

    // Submit the query
    PreparedQuery results = datastore.prepare(confQuery);

%>
```

To try out each of the queries below, you can run your local development server and visit http://localhost:<PORT_NUMBER> in your browser. Ensure that before your fire the queries, you have sufficient data that can test out these filters.

Try out the following queries:

List All Conferences in London

```
Query.Filter cityFilter = new FilterPredicate("city", FilterOperator.EQUAL, "London");
confQuery.setFilter(cityFilter);
```

Conferences having maxAttendees of 50 (Equality Filter)

```
long threshold = 50;
Query.Filter maxAttendeesFilter = new FilterPredicate("maxAttendees",
FilterOperator.EQUAL, threshold);
confQuery.setFilter(maxAttendeesFilter);
```

Conferences in "London" AND "maxAttendees" = 50

```
long threshold = 50;
Query.Filter cityFilter = new FilterPredicate("city", FilterOperator.EQUAL, "London");
Query.Filter maxAttendeesFilter = new FilterPredicate("maxAttendees",
FilterOperator.EQUAL, threshold);
Query.Filter comboFilter = CompositeFilterOperator.and(cityFilter, maxAttendeesFilter);
confQuery.setFilter(comboFilter);
```

Conferences having maxAttendees of 50 or more (InEquality Filter)

```
long threshold = 50;
Query.Filter maxAttendeesFilter = new FilterPredicate("maxAttendees",
FilterOperator.GREATER_THAN_OR_EQUAL, threshold);
confQuery.setFilter(maxAttendeesFilter);
```

If you try to execute the above Query, you should **see an error**. This is because we are applying an inequality filter on an attribute (maxAttendees) that was not originally used to sort i.e. title

To make it work, change the original Sort Order to the following:

```
confQuery.addSort("maxAttendees",SortDirection.DESCENDING)
```

To build and run the Java App Engine project, execute the following Maven commands from the command line / terminal. These commands should be executed from the **cp300-gae-datastore-java** folder i.e. where your pom.xml file is present:

1. To build the project

   ```
   mvn package
   ```

2. To run the application in the local devserver :

   ```
   mvn appengine:devserver
   ```

Python Instructions

We need to make changes to **datatore101.py** file to see the various Queries in action. You will need to replace the portion of code highlighted in bold below with the snippets of code given for each of the queries.

```
def get(self):
    conference_values = {"conferences":Conference.query().order(Conference.title)}
```

To try out each of the queries below, you can run the application via the Python local development server and visit http://localhost:<PORT_NUMBER> in your browser. Ensure that before your fire the queries, you have sufficient data that can test out these filters.

Try out the following queries:

List All Conferences in London

```
query = Conference.query().order(Conference.title)
filter_query = query.filter(Conference.city == 'London').fetch(10)
conference_values = {"conferences":filter_query}
```

Conferences having maxAttendees of 50 (Equality Filter)

```
query = Conference.query().order(Conference.title)
filter_query = query.filter(Conference.maxAttendees == 50).fetch(10)
conference_values = {"conferences":filter_query}
```

Conferences in "London" AND "maxAttendees" = 50

```
query = Conference.query().order(Conference.title)
filter_query = query.filter(Conference.city=='London',Conference.maxAttendees ==
50).fetch(10)
conference_values = {"conferences":filter_query}
```

Conferences having maxAttendees of 50 or more (InEquality Filter)

```
query = Conference.query().order(Conference.title)
filter_query = query.filter(Conference.maxAttendees >= 50).fetch(10)
conference_values = {"conferences":filter_query}
```

If you try to execute the above Query, you should **see an error**. This is because we are applying an inequality filter on an attribute (maxAttendees) that was not originally used to sort i.e. title

To make it work, change the original Sort Order to the following:

```
query = Conference.query().order(Conference.maxAttendees)
filter_query = query.filter(Conference.maxAttendees >= 50).fetch(10)
conference_values = {"conferences":filter_query}
```

Run the Python Application locally via your IDE or if you prefer via the terminal/ command line using the following:

```
dev_appserver.py /<your_python_app_dir>
```

Investigate the Indexes (Optional)

In this section, you will review index files in the code, review indexes on the development server and on App Engine.

Java (review the Datastore index file)

1. Review datastore-indexes-auto.xml in war/WEB-INF/appengine-generated/

Python (review the Datastore index file)

1. Review index.yaml.

Review the indexes in the Admin Console

Next you will review the indexes on the development server on localhost, then on a deployed App Engine application.

Review the indexes on localhost

1. Go to the admin console for your application on your **localhost**.
2. View the indexes.
   - Java: In the Datastore Viewer, select the Show Indexes link.
   - Python: Select the Datastore Indexes link.

Review the indexes on App Engine

1. Deploy the application to App Engine.
2. Go to the Admin Console for your application on appengine.google.com.
3. In the Data section, select Datastore Indexes.
   You should see the indexes that have been created.
   The indexes that have finished building are shown as SERVING.
   If they are still building, they are shown as BUILDING.

Optional Exercise:

Write a Query that requires a custom index and deploy it directly to App Engine without running it locally. Try to visit the page and check out what happens. It will give an exception that the index is not present. To remedy that you will either have to run the app locally first so that the index files are generated for you or create the index entry on your own. For details, see Configuring Indexes (Java) (Python).

Deploy an Application to App Engine directly with a Query that requires a custom index.

Additional Resources

- Datastore Queries (Java) (Python)
- Datastore Indexes (Java) (Python)
- Configuring Indexes (Java) (Python)
- How Index Building Works
- Mastering the Datastore series of articles
- How entities are indexed and stored

Summary

In this exercise, you learned how to use multiple Queries in your application. You also learnt about datastore indexes in your application.