Configure Search in the Azure Preview Portal

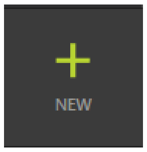
Microsoft Azure Search (Public Preview) is available in the new Preview Portal. As an administrator, you can add Search service to an existing subscription at no cost when choosing the shared service, or at a reduced rate when opting in for dedicated resources. This article has the following sections:

- Start with the free service
- Upgrade to dedicated resources
- Test service operations
- Explore Search service configuration pages
- Try these tutorials next

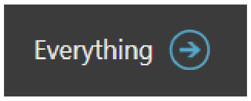
Start with the free service

Subscribers automatically get free access to a shared, multitenant Search service that you can use for learning purposes, proof-of-concept testing, or small development search projects. Sign up for the free version using these steps.

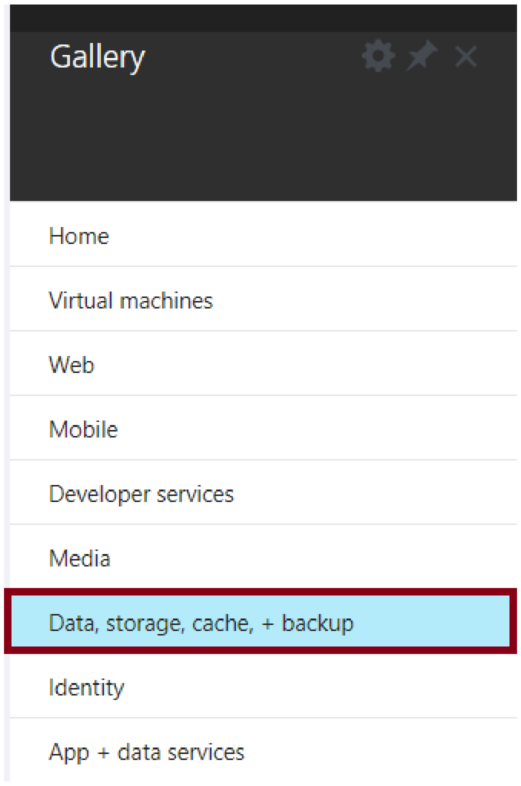
1. Sign in to Azure Preview Portal using your existing subscription. Notice that this URL takes you to the Preview Portal.
2. Click **New** at the bottom of the page.



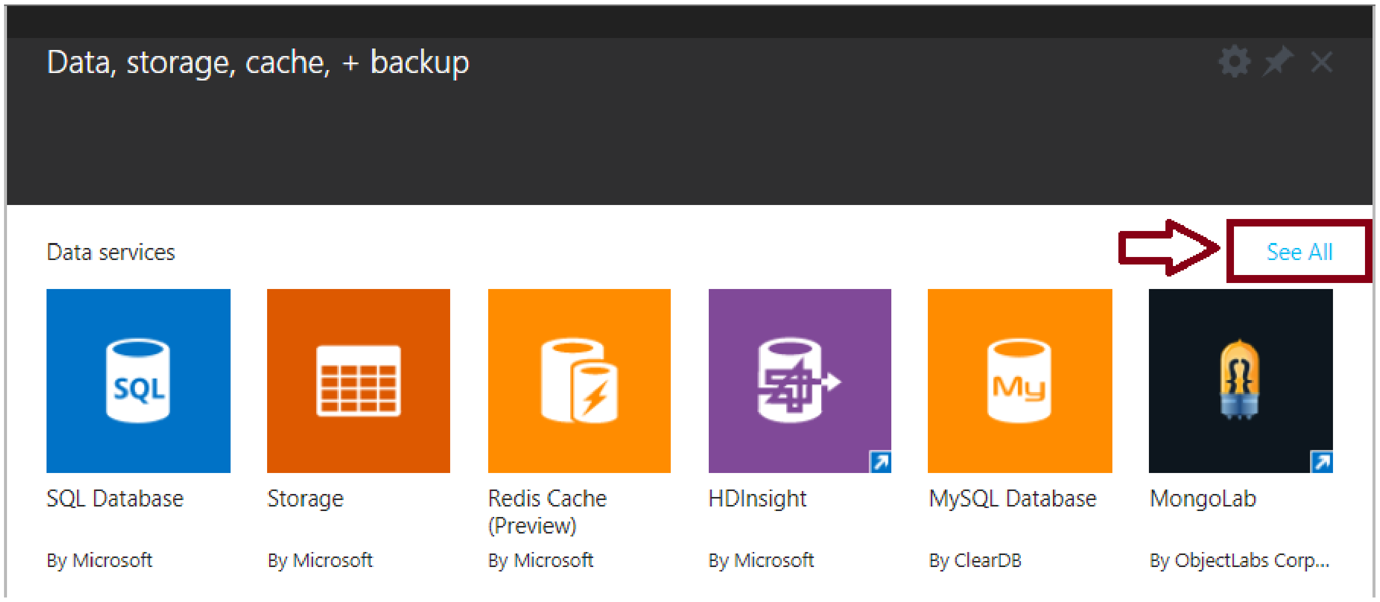
3. Click **Everything** at the top of the page.



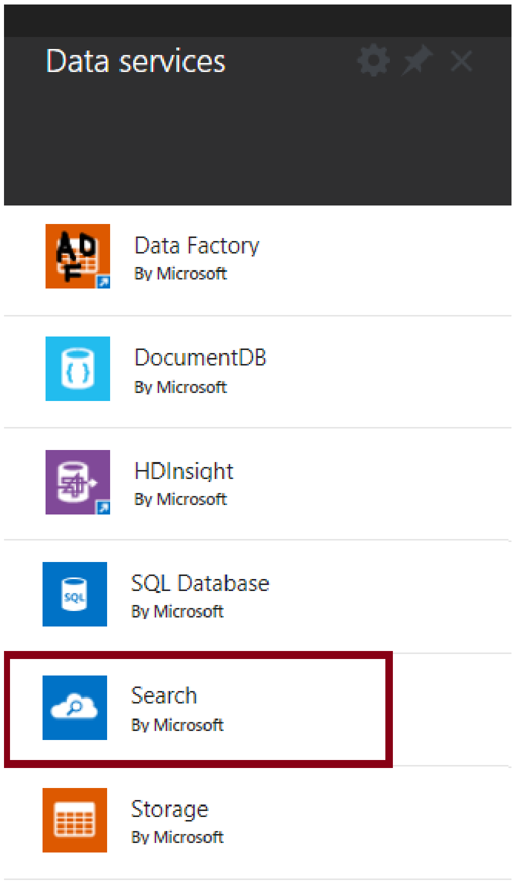
4. From the Gallery, click **Data, storage, cache, + backup**.



5. Click **See All** to expand the list of all data-related services.

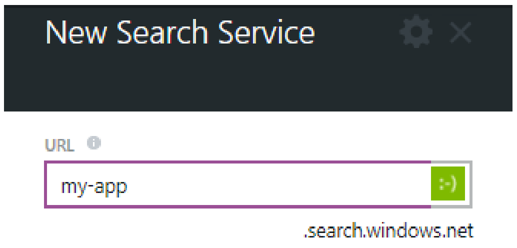


6. From Data services, click **Search**.



7. At the bottom of the Search page, click **CREATE**.

8. Type a lower-case service name to use in the service URL, avoiding dashes, spaces, and staying within the 15 character string limit.

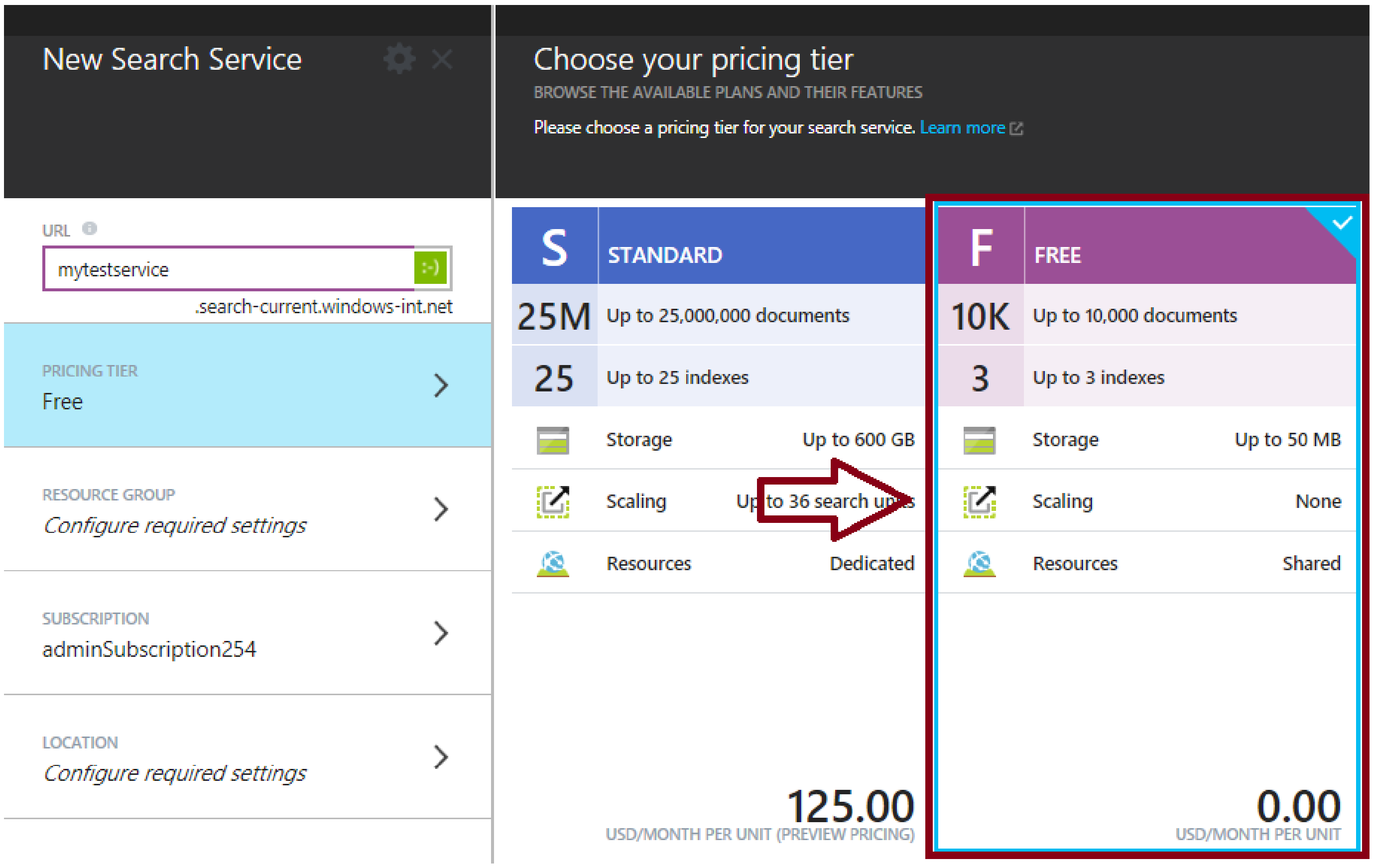

New Search Service

URL

my-app

.search.windows.net

9. Click the arrow in **Pricing Tier** to select a pricing option. Choose **FREE** and then click **SELECT** at the bottom of the page. The free version offers enough capacity to try out tutorials and write proof-of-concept code, but is not intended for production applications.


New Search Service

URL

mytestservice

.search-current.windows-int.net

PRICING TIER
Free

RESOURCE GROUP
Configure required settings

SUBSCRIPTION
adminSubscription254

LOCATION
Configure required settings

Choose your pricing tier

BROWSE THE AVAILABLE PLANS AND THEIR FEATURES

Please choose a pricing tier for your search service. Learn more

S STANDARD		F FREE	
25M	Up to 25,000,000 documents	10K	Up to 10,000 documents
25	Up to 25 indexes	3	Up to 3 indexes
Storage	Up to 600 GB	Storage	Up to 50 MB
Scaling	Up to 36 search units	Scaling	None
Resources	Dedicated	Resources	Shared
125.00	USD/MONTH PER UNIT (PREVIEW PRICING)	0.00	USD/MONTH PER UNIT

10. Click the arrow in **Resource Group** to pick an existing group or create a new one. Resource groups are containers for services and resources used for a common purpose. For example, if you're building a custom search application based on Azure Search, Azure Web sites, Azure BLOB storage, you could create a resource group that keeps these services together in the portal management pages.

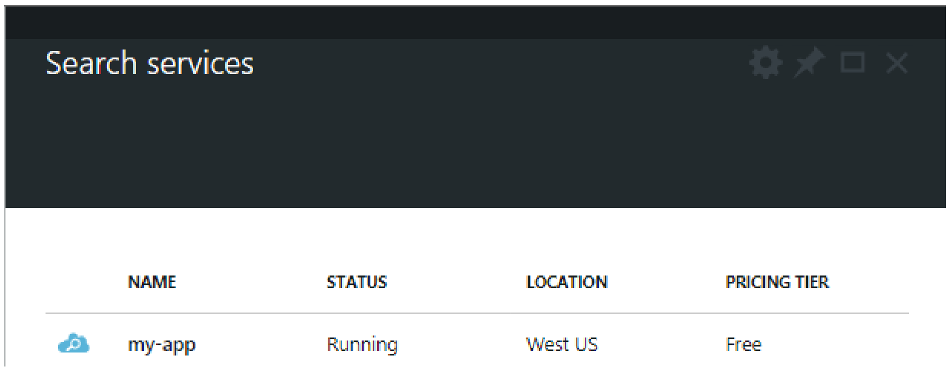
11. Click the arrow in **Subscription** if you have multiple subscriptions and you want to use a different subscription for this search service.

12. Click the arrow in **Location** to choose a data center region. In this preview, only West US is available. Later, when other regions are online, choose one region for the service you are creating. Distributing resources across multiple data centers will not be a supported configuration.

13. Click **CREATE** to provision the service. Notice that **CREATE** is enabled only after you fill in all required values.

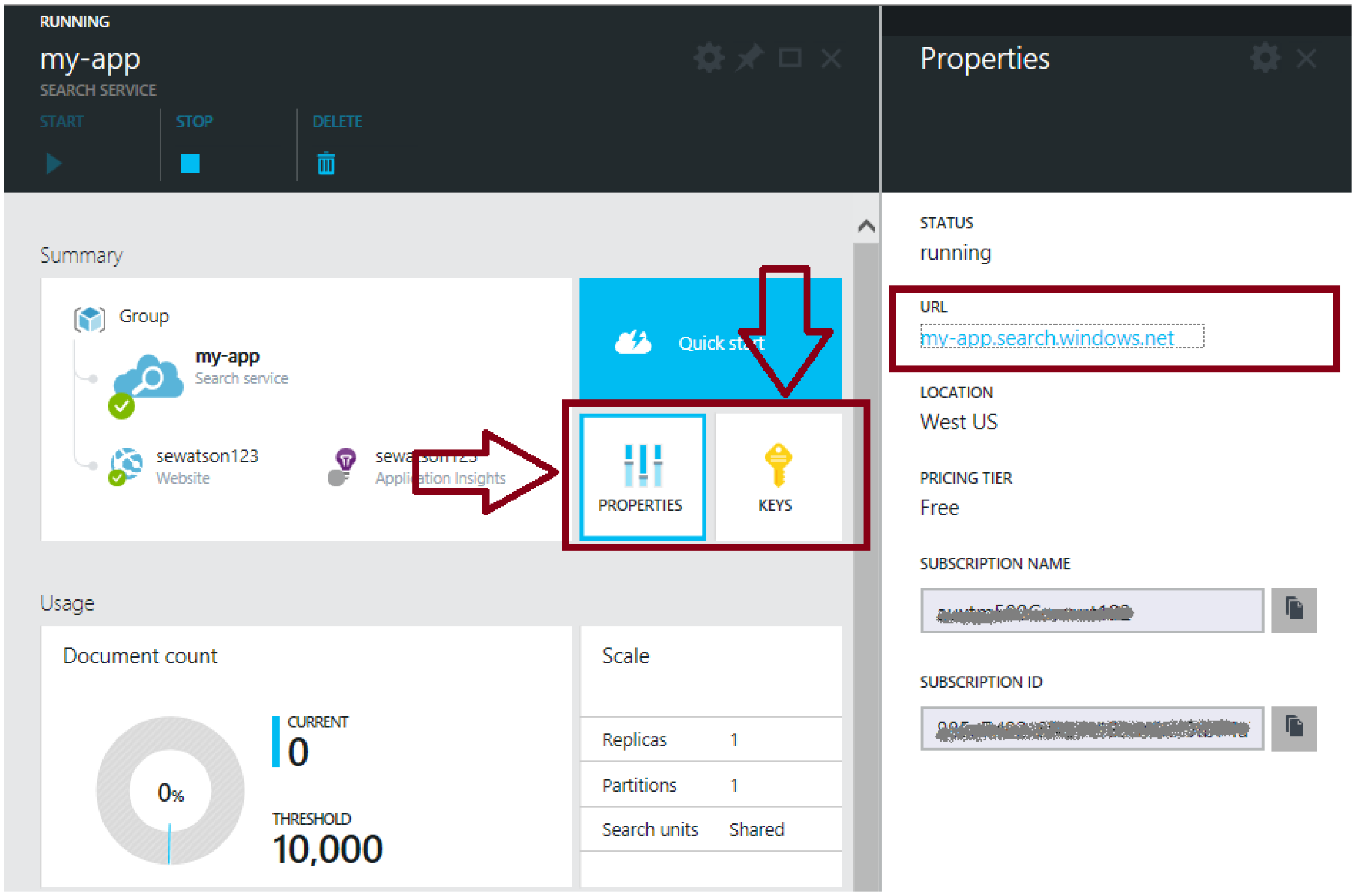
In a few minutes, the service is created. You can return to the configuration settings to get the URL or api-keys. Connections to your Search service requires that you have both the URL and an api-key to authenticate the call. Here's how to quickly find these values:

1. Go to **Browse** | **Everything** | **Data, storage, cache, + backup** | **See All** | **Search Services**, Click your search service to open the service dashboard.


Search services

NAME	STATUS	LOCATION	PRICING TIER
my-app	Running	West US	Free

2. On the service dashboard, you'll see tiles for **PROPERTIES** and **KEYS**, and usage information that shows resource usage at at glance.


RUNNING
my-app
SEARCH SERVICE

Document count
0%
CURRENT
0
THRESHOLD
10,000

Scale

Replicas	1
Partitions	1
Search units	Shared

Properties

STATUS
running

URL
my-app.search.windows.net

LOCATION
West US

PRICING TIER
Free

SUBSCRIPTION NAME
[redacted]

SUBSCRIPTION ID
[redacted]

PROPERTIES contains the service URL.

KEYS contains the api-keys used for authentication.

USAGE shows the document count, available resources, and storage limits.

Continue on Test service operations for instructions on how to connect to the service using these values.

Upgrade to dedicated resources

Dedicated resources are machines in an Azure data center that can be used only by you. Search workloads require storage and service resources. You can optimize service configuration to use more of whatever resource is the most important to your scenario.

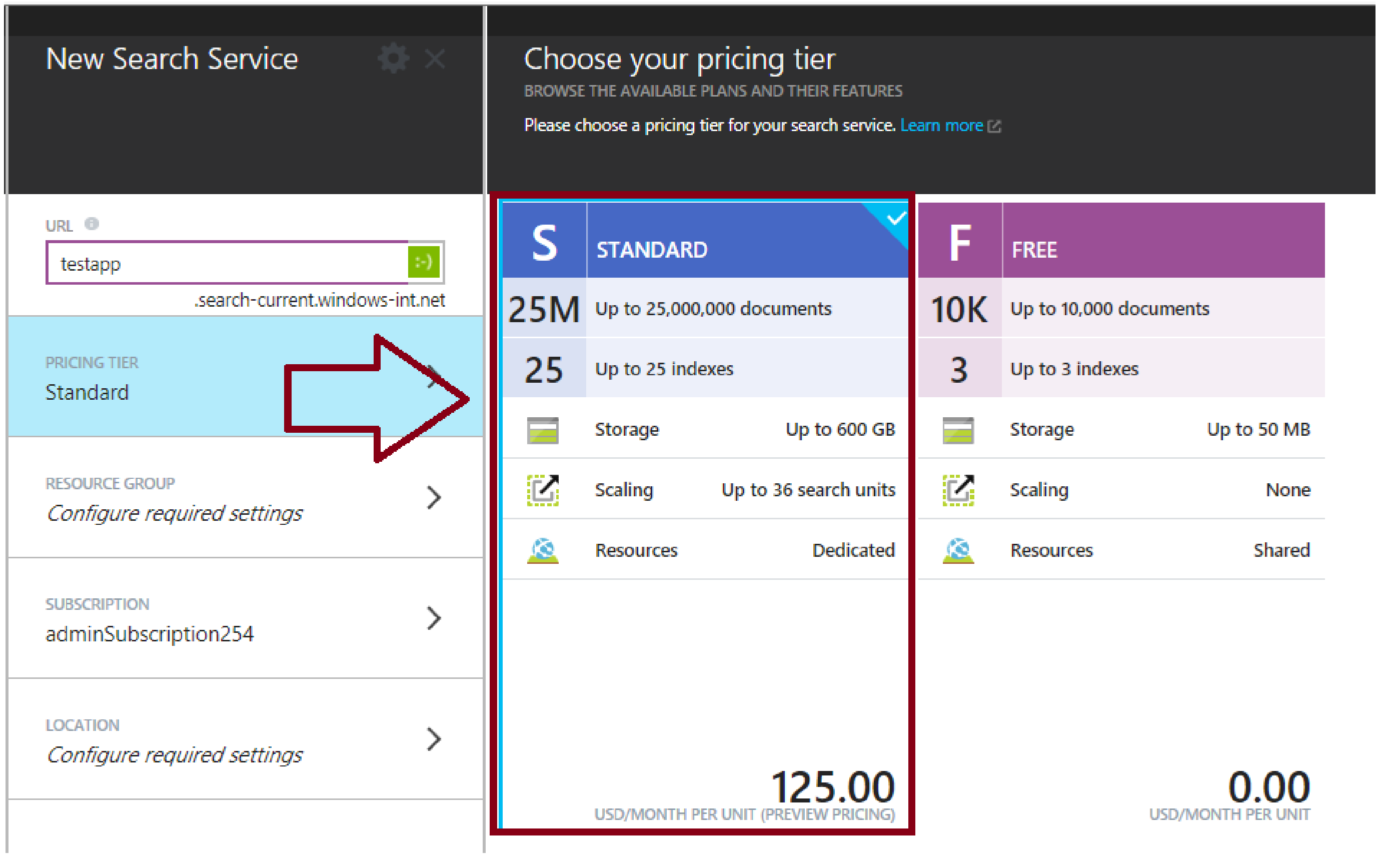
Having dedicated resources will give you more scale and better performance, but not additional features. Both shared and dedicated search offer the same features.

To use dedicated search, create a new Search service, choosing the Standard pricing tier. Notice that upgrade is not an in-place upgrade of the free version. Switching to dedicated storage, with its potential for scale, requires a new service. You will need to reload the indexes and documents used by your search application.

Setting up dedicated machines can take a while (15 minutes or longer) depending on how many replicas and partitions you use.

Step 1 - Create a new service with Pricing Tier set to Standard

1. Sign in to Azure Preview Portal using your existing subscription.
2. Click **New** at the bottom of the page.
3. Click **Everything** at the top of the page.
4. From the Gallery, click **Data, storage, cache, + backup**.
5. Click **See All** to expand the list of all data-related services.
6. From Data services, click **Search**.
7. At the bottom of the Search page, click **CREATE**.
8. Type a lower-case service name to use in the service URL, avoiding dashes, spaces, and staying under the 15 character string limit.
9. Click the arrow in **Pricing Tier** to select a pricing option. Choose **STANDARD** and then click **SELECT** at the bottom of the page.



New Search Service

URL: testapp .search-current.windows-int.net

PRICING TIER: Standard

RESOURCE GROUP: *Configure required settings*

SUBSCRIPTION: adminSubscription254

LOCATION: *Configure required settings*

Choose your pricing tier

BROWSE THE AVAILABLE PLANS AND THEIR FEATURES

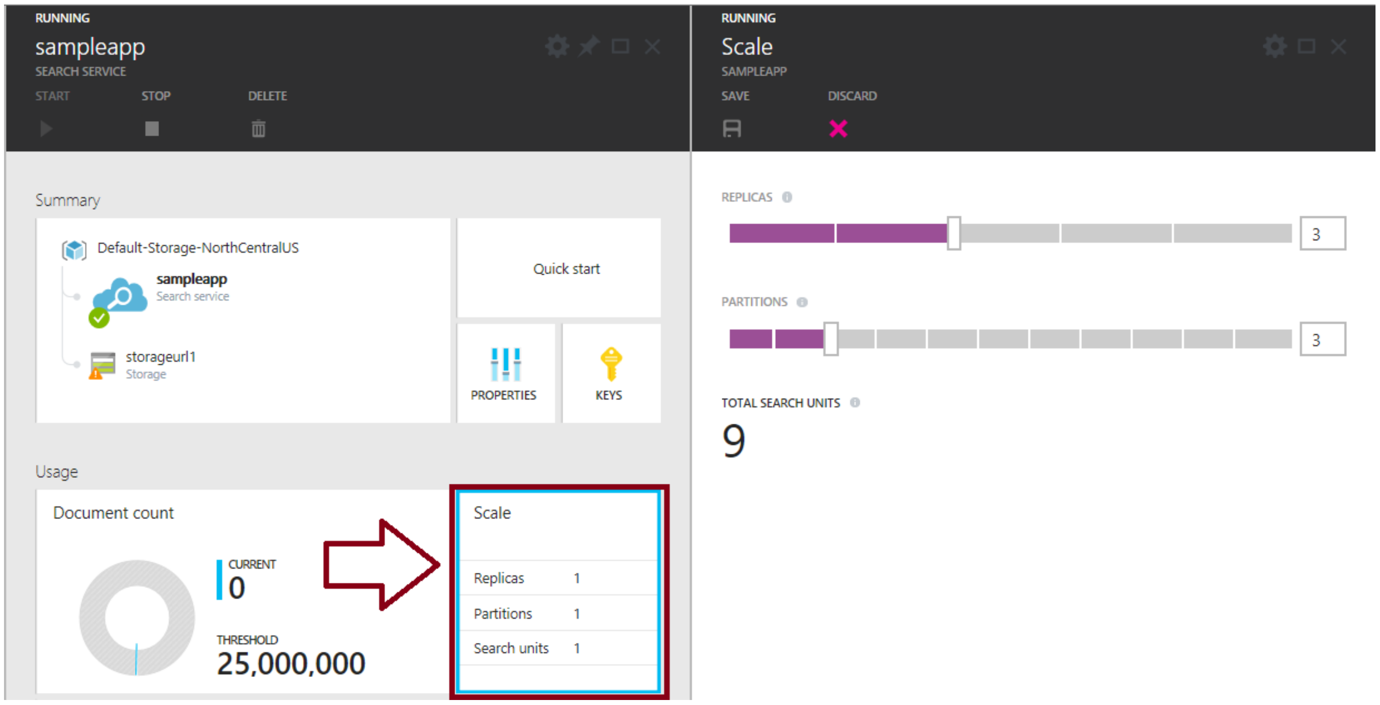
Please choose a pricing tier for your search service. Learn more

S STANDARD		F FREE	
25M	Up to 25,000,000 documents	10K	Up to 10,000 documents
25	Up to 25 indexes	3	Up to 3 indexes
Storage	Up to 600 GB	Storage	Up to 50 MB
Scaling	Up to 36 search units	Scaling	None
Resources	Dedicated	Resources	Shared
125.00	USD/MONTH PER UNIT (PREVIEW PRICING)	0.00	USD/MONTH PER UNIT

Step 2 - Adjust search units based on scale requirements

Dedicated search services start with a single search unit, but can be re-scaled at higher resource levels.

1. Once the service is created, return to the service dashboard, click the **Scale** tile.
2. Use the sliders to add replicas, partitions, or both. The total search units required to support any particular resource configuration is shown on the page. You can check Pricing Details to get the per-unit billing information.



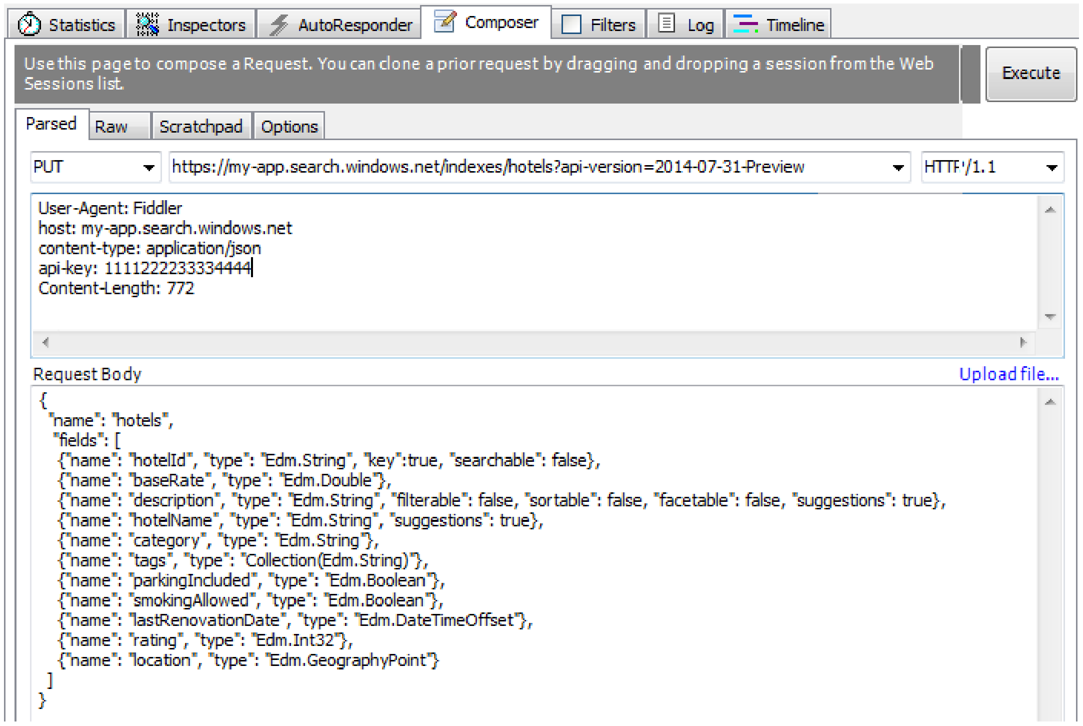
Test service operations

Confirming that your service is operational and accessible from a client application is the final step in configuring Search. This procedure uses Fiddler, available as a free download from Telerik, to issue HTTP requests and view responses. By using Fiddler, you can test the API immediately, without having to write any code.

The following procedure works for both multitenant and dedicated search. In the steps below, you'll create an index, upload documents, query the index, and then query the system for service information.

Create an index

1. Start Fiddler. On the File menu, turn off **Capture Traffic** to hide extraneous HTTP activity that is unrelated to the current task. On the Composer tab, you'll formulate a request that looks like this:


Statistics
Inspectors
AutoResponder
Composer
Filters
Log
Timeline
Use this page to compose a Request. You can clone a prior request by dragging and dropping a session from the Web Sessions list.
Execute
Parsed
Raw
Scratchpad
Options
Request Body
Upload file...

```
PUT    https://my-app.search.windows.net/indexes/hotels?api-version=2014-07-31-Preview    HTTP/1.1
```

```
User-Agent: Fiddler
host: my-app.search.windows.net
content-type: application/json
api-key: 1111222233334444
Content-Length: 772
```

```
{
 "name": "hotels",
  "fields": [
   {"name": "hotelId", "type": "Edm.String", "key":true, "searchable": false},
   {"name": "baseRate", "type": "Edm.Double"},
   {"name": "description", "type": "Edm.String", "filterable": false, "sortable": false, "facetable": false, "suggestions": true},
   {"name": "hotelName", "type": "Edm.String", "suggestions": true},
   {"name": "category", "type": "Edm.String"},
   {"name": "tags", "type": "Collection(Edm.String)"},
   {"name": "parkingIncluded", "type": "Edm.Boolean"},
   {"name": "smokingAllowed", "type": "Edm.Boolean"},
   {"name": "lastRenovationDate", "type": "Edm.DateTimeOffset"},
   {"name": "rating", "type": "Edm.Int32"},
   {"name": "location", "type": "Edm.GeographyPoint"}
  ]
}
```

2. Select **PUT**.
3. Enter a URL that specifies the service URL (which you can find on the Properties page), request attributes and the api-version. A few pointers to keep in mind:

    - Use HTTPS as the prefix
    - Request attribute is “/indexes/hotels". This tells Search to create an index named 'hotels'.
    - Api-version is lower-case, specified as ?api-version=2014-07-31-preview”.

    The full URL should look similar to the following example:

    ```
     https://my-app.search.windows.net/indexes/hotels?api-version=2014-07-31-Preview
    ```

4. Specify the request header, replacing the host and api-key (lower-case) with values that are valid for your service.

    ```
    User-Agent: Fiddler
    host: my-app.search.windows.net
    content-type: application/json
    api-key: 1111222233334444
    ```

5. In Request Body, paste in the fields that make up the index definition.

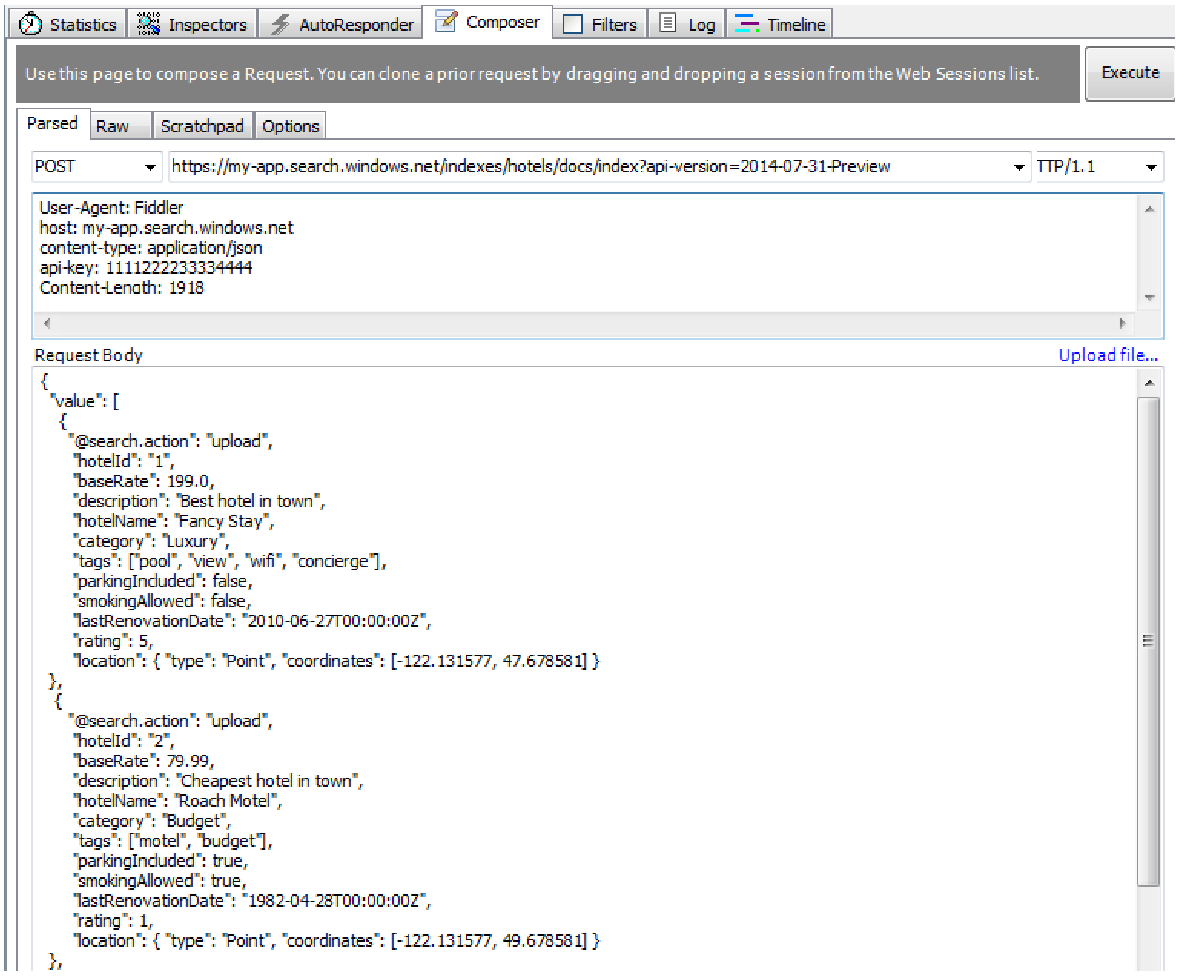
    ```
     {
    "name": "hotels",
    "fields": [
      {"name": "hotelId", "type": "Edm.String", "key":true, "searchable": false},
      {"name": "baseRate", "type": "Edm.Double"},
      {"name": "description", "type": "Edm.String", "filterable": false, "sortable": false, "facetable": false, "suggestions": tru
      {"name": "hotelName", "type": "Edm.String", "suggestions": true},
      {"name": "category", "type": "Edm.String"},
      {"name": "tags", "type": "Collection(Edm.String)"},
      {"name": "parkingIncluded", "type": "Edm.Boolean"},
      {"name": "smokingAllowed", "type": "Edm.Boolean"},
      {"name": "lastRenovationDate", "type": "Edm.DateTimeOffset"},
      {"name": "rating", "type": "Edm.Int32"},
      {"name": "location", "type": "Edm.GeographyPoint"}
     ]
    }
    ```

6. Click **Execute**.

In a few seconds, you should see an HTTP 204 response in the session list, indicating the index was created successfully. If you get HTTP 504, verify the URL specifies HTTPS. If you see HTTP 404, double-check your syntax. HTTP 400 means there is a problem with the api-key (either an invalid key or a syntax problem on the api-key entry)

Load documents

On the Composer tab, your request to post documents will look like the following. The body of the request contains the search data for 4 hotels.



Use this page to compose a Request. You can clone a prior request by dragging and dropping a session from the Web Sessions list.

POST https://my-app.search.windows.net/indexes/hotels/docs/index?api-version=2014-07-31-Preview TTP/1.1

```
User-Agent: Fiddler
host: my-app.search.windows.net
content-type: application/json
api-key: 1111222233334444
Content-Length: 1918
```

```
{
 "value": [
  {
   "@search.action": "upload",
   "hotelId": "1",
   "baseRate": 199.0,
   "description": "Best hotel in town",
   "hotelName": "Fancy Stay",
   "category": "Luxury",
   "tags": ["pool", "view", "wifi", "concierge"],
   "parkingIncluded": false,
   "smokingAllowed": false,
   "lastRenovationDate": "2010-06-27T00:00:00Z",
   "rating": 5,
   "location": { "type": "Point", "coordinates": [-122.131577, 47.678581] }
 },
  {
   "@search.action": "upload",
   "hotelId": "2",
   "baseRate": 79.99,
   "description": "Cheapest hotel in town",
   "hotelName": "Roach Motel",
   "category": "Budget",
   "tags": ["motel", "budget"],
   "parkingIncluded": true,
   "smokingAllowed": true,
   "lastRenovationDate": "1982-04-28T00:00:00Z",
   "rating": 1,
   "location": { "type": "Point", "coordinates": [-122.131577, 49.678581] }
 },
```

1. Select **POST**.

2. Enter a URL that starts with HTTPS, followed by your service URL, followed by “/indexes/<'indexname'>/docs/index?api-version=2014-07-31-preview”. The full URL should look similar to the following example:

```
https://my-app.search.windows.net/indexes/hotels/docs/index?api-version=2014-07-31-Preview
```

3. Request Header should be the same as before. Remember that you replaced the host and api-key with values that are valid for your service.

```
User-Agent: Fiddler
host: my-app.search.windows.net
content-type: application/json
api-key: 1111222233334444
```

4. The Request Body contains four documents to be added to the hotels index.

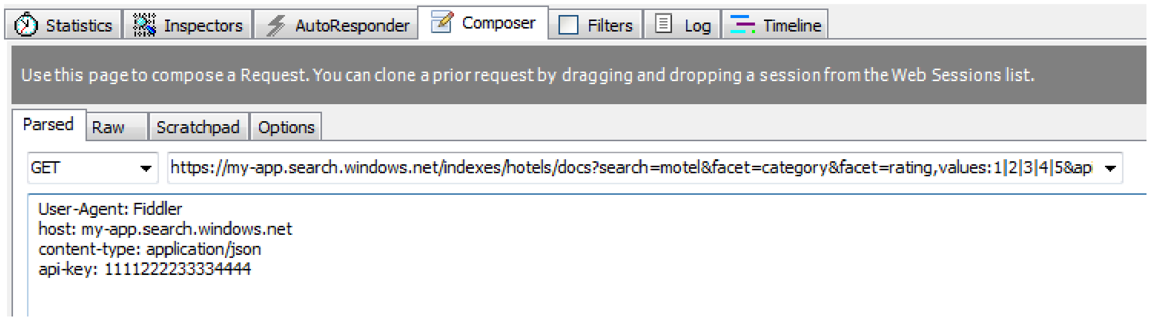
```
{
"value": [
{
    "@search.action": "upload",
    "hotelId": "1",
    "baseRate": 199.0,
    "description": "Best hotel in town",
    "hotelName": "Fancy Stay",
    "category": "Luxury",
    "tags": ["pool", "view", "wifi", "concierge"],
    "parkingIncluded": false,
    "smokingAllowed": false,
    "lastRenovationDate": "2010-06-27T00:00:00Z",
    "rating": 5,
    "location": { "type": "Point", "coordinates": [-122.131577, 47.678581] }
  },
  {
    "@search.action": "upload",
    "hotelId": "2",
    "baseRate": 79.99,
    "description": "Cheapest hotel in town",
    "hotelName": "Roach Motel",
    "category": "Budget",
    "tags": ["motel", "budget"],
    "parkingIncluded": true,
    "smokingAllowed": true,
    "lastRenovationDate": "1982-04-28T00:00:00Z",
    "rating": 1,
    "location": { "type": "Point", "coordinates": [-122.131577, 49.678581] }
  },
  {
    "@search.action": "upload",
    "hotelId": "3",
    "baseRate": 279.99,
    "description": "Surprisingly expensive",
    "hotelName": "Dew Drop Inn",
    "category": "Bed and Breakfast",
    "tags": ["charming", "quaint"],
    "parkingIncluded": true,
    "smokingAllowed": false,
    "lastRenovationDate": null,
    "rating": 4,
    "location": { "type": "Point", "coordinates": [-122.33207, 47.60621] }
  },
  {
    "@search.action": "upload",
    "hotelId": "4",
    "baseRate": 220.00,
    "description": "This could be the one",
    "hotelName": "A Hotel for Everyone",
    "category": "Basic hotel",
    "tags": ["pool", "wifi"],
    "parkingIncluded": true,
    "smokingAllowed": false,
    "lastRenovationDate": null,
    "rating": 4,
    "location": { "type": "Point", "coordinates": [-122.12151, 47.67399] }
  }
 ]
}
```

5. Click **Execute**.

In a few seconds, you should see an HTTP 200 response in the session list. This indicates the documents were created successfully. If you get a 207, at least one document failed to upload. If you get a 404, you have a syntax error in either the header or body of the request.

Query the index

Now that an index and documents are loaded, you can issue queries against them. On the Composer tab, a GET command that queries your service will look similar to the following:



Use this page to compose a Request. You can clone a prior request by dragging and dropping a session from the Web Sessions list.

https://my-app.search.windows.net/indexes/hotels/docs?search=motel&facet=category&facet=rating,values:1|2|3|4|5&ap

```
User-Agent: Fiddler
host: my-app.search.windows.net
content-type: application/json
api-key: 1111222233334444
```

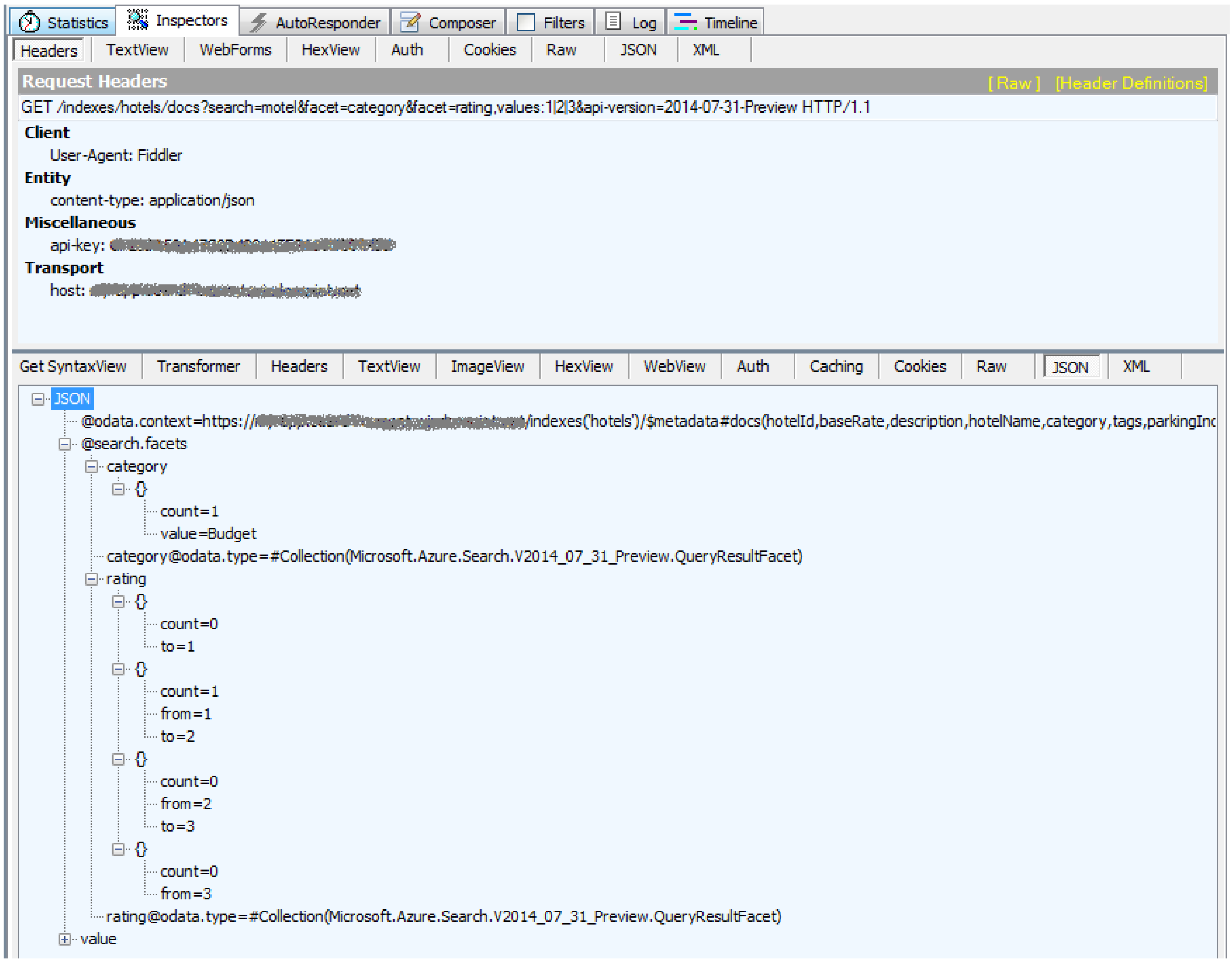
1. Select **GET**.
2. Enter a URL that starts with HTTPS, followed by your service URL, followed by "/indexes//docs?", followed by query parameters. By way of example, use the following URL, replacing the sample host name with one that is valid for your service.

   ```
   https://my-app.search.windows.net/indexes/hotels/docs?search=motel&facet=category&facet=rating,values:1|2|3|4|5&api-version=20
   ```

   This query searches on the term "motel" and retrieves facet categories for ratings.
3. Request Header should be the same as before. Remember that you replaced the host and api-key with values that are valid for your service.

   ```
   User-Agent: Fiddler
   host: my-app.search.windows.net
   content-type: application/json
   api-key: 1111222233334444
   ```

The response code should be 200, and the response output should look similar to the following illustration.



Request Headers [Raw] [Header Definitions]

GET /indexes/hotels/docs?search=motel&facet=category&facet=rating,values:1|2|3&api-version=2014-07-31-Preview HTTP/1.1

Client
User-Agent: Fiddler
Entity
content-type: application/json
Miscellaneous
api-key: [redacted]
Transport
host: [redacted]

```
JSON
  @odata.context=https://[redacted]/indexes('hotels')/$metadata#docs(hotelId,baseRate,description,hotelName,category,tags,parkingInc
  @search.facets
    category
      {}
        count=1
        value=Budget
    category@odata.type=#Collection(Microsoft.Azure.Search.V2014_07_31_Preview.QueryResultFacet)
    rating
      {}
        count=0
        to=1
      {}
        count=1
        from=1
        to=2
      {}
        count=0
        from=2
        to=3
      {}
        count=0
        from=3
    rating@odata.type=#Collection(Microsoft.Azure.Search.V2014_07_31_Preview.QueryResultFacet)
  value
```

The following example query is from the Search API documentation. These queries include spaces, which are not allowed in Fiddler. Replace each space with a + character before pasting in the query string:

Before spaces are replaced:

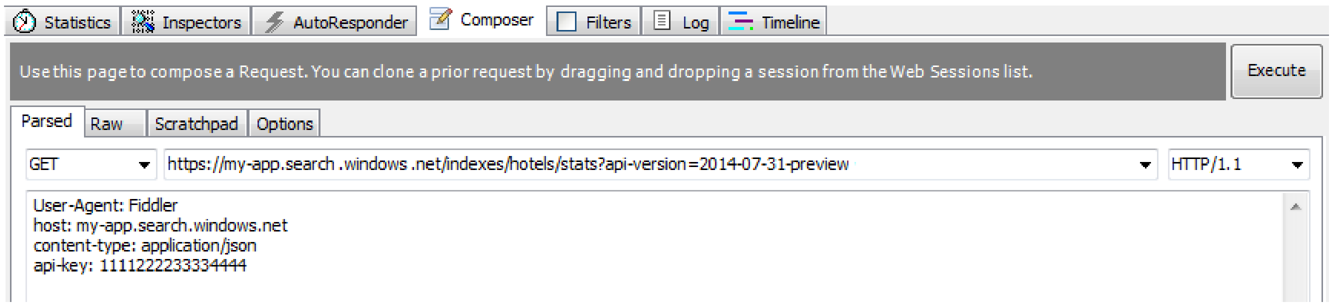
```
GET /indexes/hotels/docs?search=*&$orderby=lastRenovationDate desc&api-version=2014-07-31-Preview
```

After spaces are replaced with +:

```
GET /indexes/hotels/docs?search=*&$orderby=lastRenovationDate+desc&api-version=2014-07-31-Preview
```

Query the system

You can also query the system to get document counts and storage consumption. On the Composer tab, your request will look like the following, and the response will return a count for the number of documents and space used.



Use this page to compose a Request. You can clone a prior request by dragging and dropping a session from the Web Sessions list.

GET https://my-app.search.windows.net/indexes/hotels/stats?api-version=2014-07-31-preview HTTP/1.1

```
User-Agent: Fiddler
host: my-app.search.windows.net
content-type: application/json
api-key: 1111222233334444
```

1. Select **GET**.

2. Enter a URL that includes your service URL, followed by "/indexes/hotels/stats?api-version=2014-07-31-preview":

```
https://my-app.search.windows.net/indexes/hotels/stats?api-version=2014-07-31-preview
```

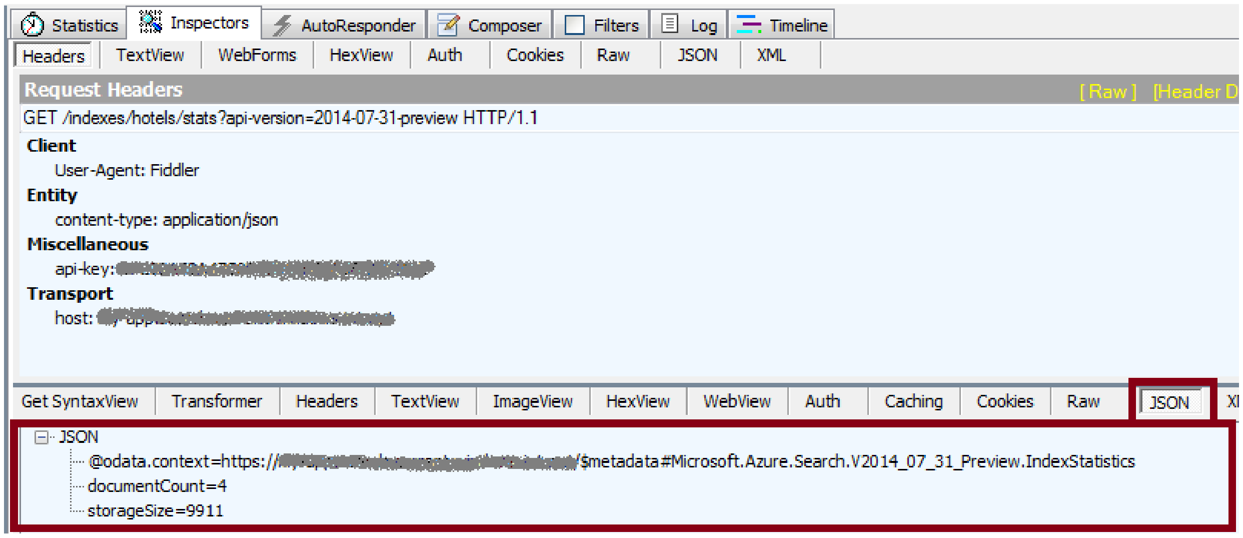
3. Specify the request header, replacing the host and api-key with values that are valid for your service.

```
User-Agent: Fiddler
host: my-app.search.windows.net
content-type: application/json
api-key: 1111222233334444
```

4. Leave the request body empty.

5. Click **Execute**. You should see an HTTP 200 status code in the session list. Select the entry posted for your command.

6. Click the Inspectors tab | Headers, and select the JSON format. You should see the document count and storage size (in KB).



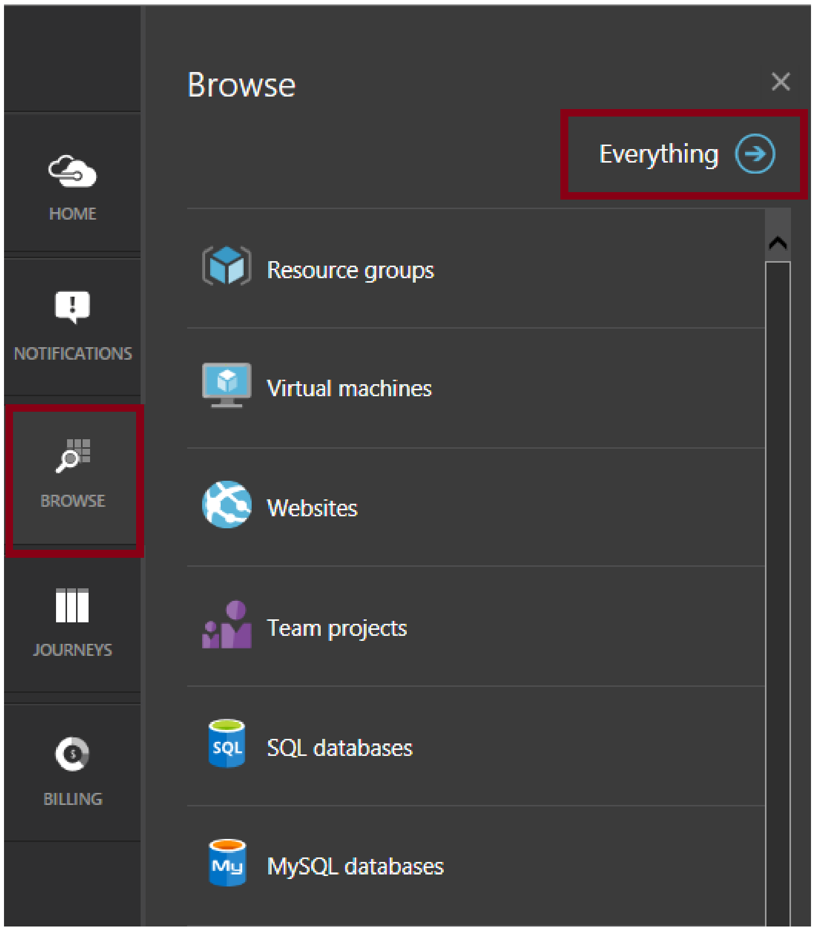
```
GET /indexes/hotels/stats?api-version=2014-07-31-preview HTTP/1.1
Client
    User-Agent: Fiddler
Entity
    content-type: application/json
Miscellaneous
    api-key: [redacted]
Transport
    host: [redacted]
```

```
JSON
    @odata.context=https://[redacted]/$metadata#Microsoft.Azure.Search.V2014_07_31_Preview.IndexStatistics
    documentCount=4
    storageSize=9911
```

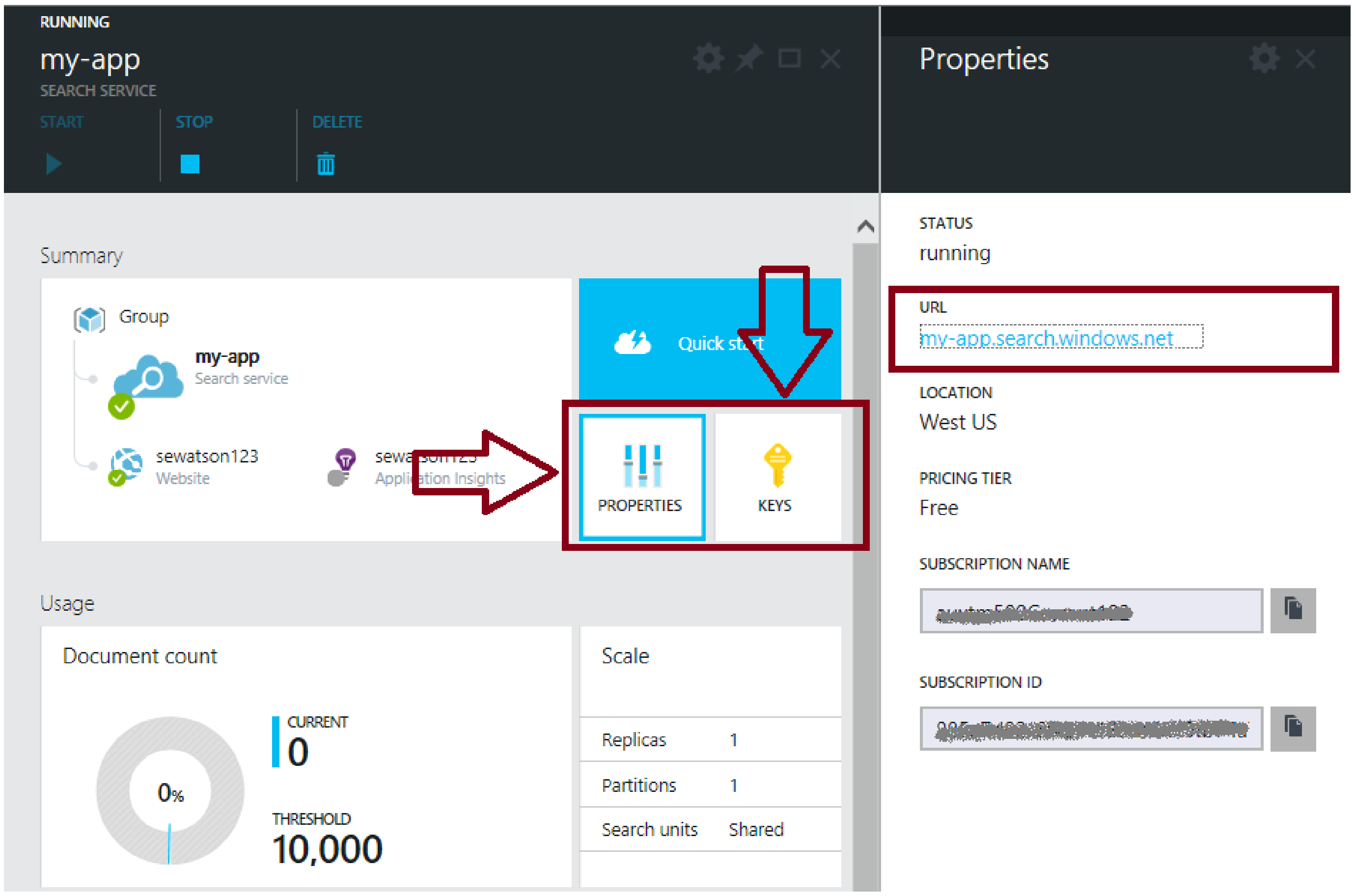
Explore Search service configuration pages

If you're inheriting a service created by someone else or need some help with page navigation, follow these steps to locate the service dashboard.

1. Sign in to Azure Preview Portal using your existing subscription.
2. Click **Browse** | **Everything**.



3. Choose **Search services** from the list. You should see a list of all the Search services created under your subscriptions.
4. Click a service to open its dashboard. Notice that **Start**, **Stop**, and **Delete** commands are at the top. The service dashboard includes tiles for viewing Properties, Keys, and a Quick Start with links to information and instructions. Scroll down to view usage.
5. Click **PROPERTIES**. Notice that the Properties page opens to the right. The service URL is at the top of the page. To get api-keys used to authenticate to the service, click **KEYS**.



Replicas	1
Partitions	1
Search units	Shared

STATUS
running

URL
my-app.search.windows.net

LOCATION
West US

PRICING TIER
Free

SUBSCRIPTION NAME
[redacted]

SUBSCRIPTION ID
[redacted]

Try it out

Ready for the next step? The following links take you to additional material that shows you how to build and manage search applications that use Azure Search.

Create your first azure search solution

Manage your search solution in Microsoft Azure

Azure Search on MSDN

Search REST API on MSDN