Lab: Using the Cloudant NoSQL database

After you complete this lab, you should understand:

- Useful features in Cloudant NoSQL database
- How to create an instance of a Cloudant database
- How to add views
- How to make RESTful calls to Cloudant by using view and list functions

IBM Cloudant

- IBM Cloudant is a NoSQL database and an extension of Apache CouchDB. It stores JSON documents, uses JavaScript for MapReduce indexes, and HTTP RESTful APIs.
- Cloudant is designed for the Internet: it’s optimized for handling heavy workloads of concurrent reads and writes in the cloud, has a flexible schema, and massive scaling for fast-growing web and mobile applications.
- Cloudant provides:
  - Fully managed DaaS (database as a service) to offload administration and maintenance from end users
  - More massive horizontal scaling than relational databases to compensate for fast growing data sets and surges in number of concurrent users
  - High availability with offline and online access capability through integrated replication and sync
  - Economical architecture since data is distributed, and database is deployed/operated on clusters of servers in the cloud
  - Flexible schema and intuitive data structures that help you build new features into applications without locking the database
  - Ability to index and query document content, range look-ups, and execute analytical queries through MapReduce functions

When you shouldn’t use Cloudant database

- When applications require transaction ACID capabilities with relationships spanning multiple documents.

Prerequisites

Download code from Github

1. Go to Github at https://github.com/ibmecod/cloudantLab.
2. Click **Download ZIP**.
3. Extract the cloudantLB-master.zip file into your download folder.

Instructions for Windows users using cURL

You should use the version of cURL included in the downloaded ZIP file because it includes SSL support that is required for this lab. (cURL is often installed on Windows without SSL support.)

To install cURL with SSL support:

1. If you don't have cURL installed, skip to step 3. Otherwise, verify that your installed cURL has SSL support by running the command `curl https://www.google.com`. If there are no errors, skip the rest of these instructions. If you encounter an error, go to step 2.
2. Uninstall other cURL versions or remove them from your PATH variable.
3. As an administrator, install cURL from the downloaded ZIP file named ***curl-7.43.0-win64-local.msi .***

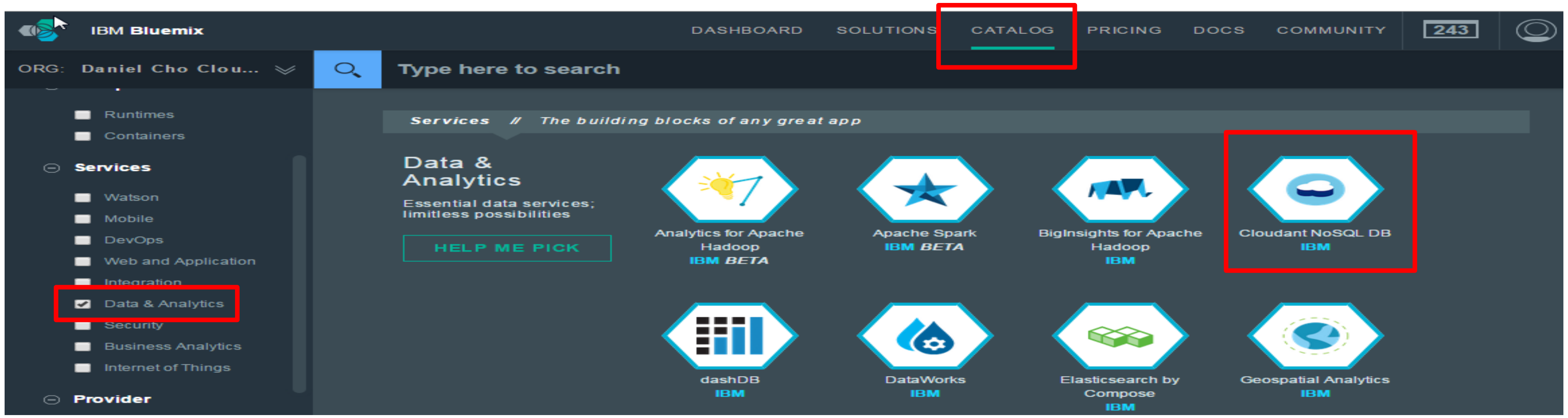
   Important: You must install cURL as an administrator; otherwise, the installer will not modify the PATH environment variable. Use an administrator command prompt and run the MSI file from that prompt.

4. Verify that `curl.exe` is now in your PATH variable and is configured for SSL by opening a new command prompt and running the command `curl https://www.google.com`. The command should run without error.

Create a space in Bluemix

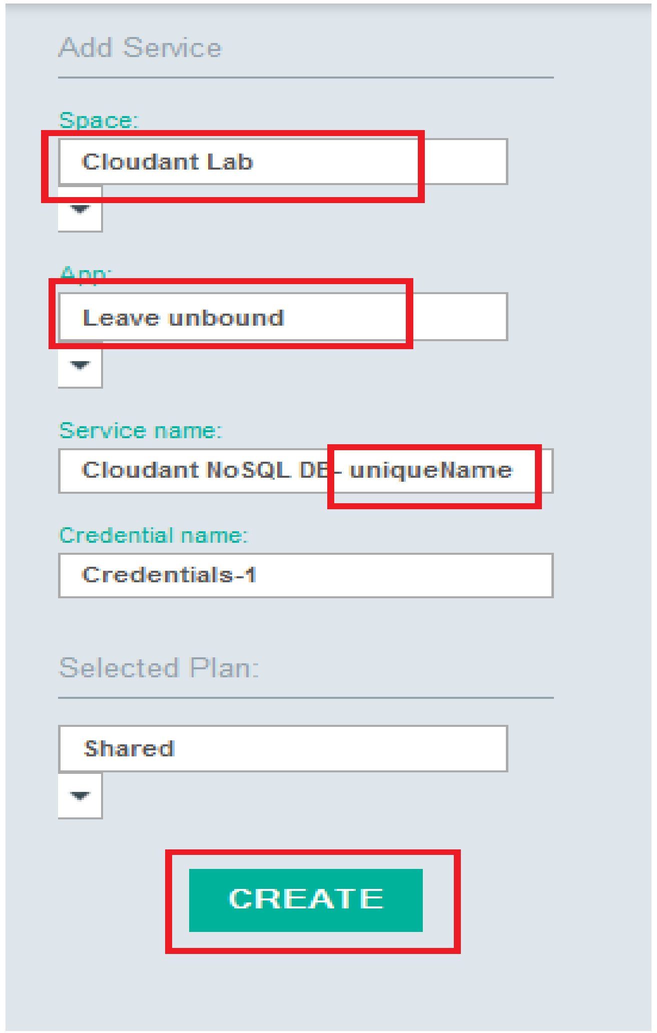
For simplicity, create a space where you can create an unbounded Cloudant NoSQL DB instance.

1. If you haven't done so, create a Bluemix account.
2. Log in to Bluemix at https://bluemix.net.
3. From the Bluemix dashboard, click **Create a Space** and enter the name `Cloudant Lab`.
4. Click **Catalog**. In the left navigation, select **Data & Analytics** under **Services**.
5. Select **Cloudant NoSQL DB** to create an instance.



Create an instance of CloudantDB

7. Select **Leave unbound**, which means that the service is not yet bound to any application.
8. Enter a unique name for the service.
9. Use the default for other entries.
10. Click **CREATE**. You'll see the following Cloudant NoSQL instance:


Add Service

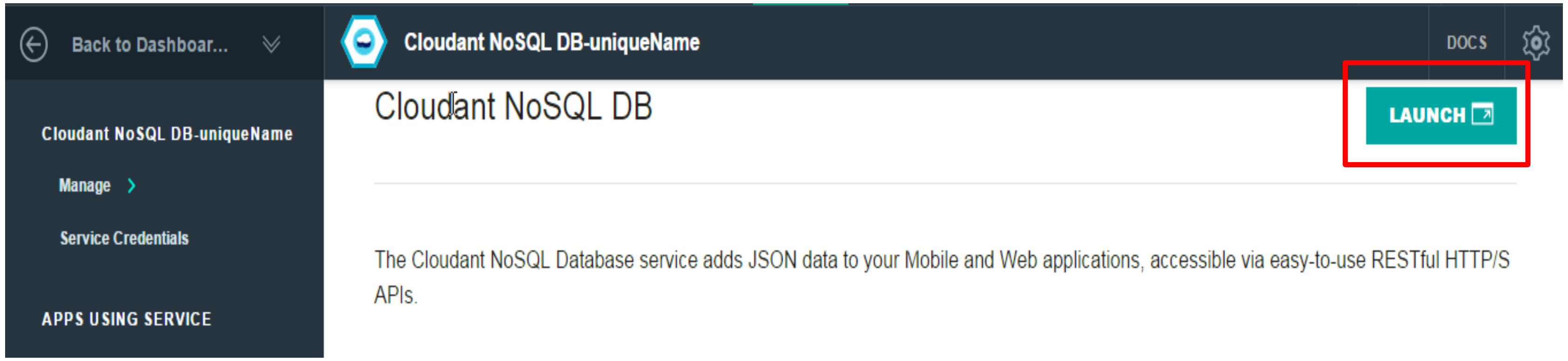
Space: Cloudant Lab

App: Leave unbound

Service name: Cloudant NoSQL DB- uniqueName

Credential name: Credentials-1

Selected Plan: Shared



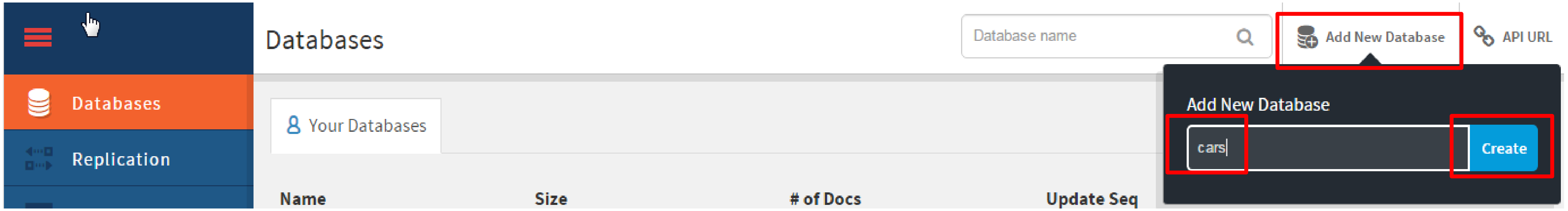
Cloudant NoSQL DB

The Cloudant NoSQL Database service adds JSON data to your Mobile and Web applications, accessible via easy-to-use RESTful HTTP/S APIs.

11. Click **Launch**.

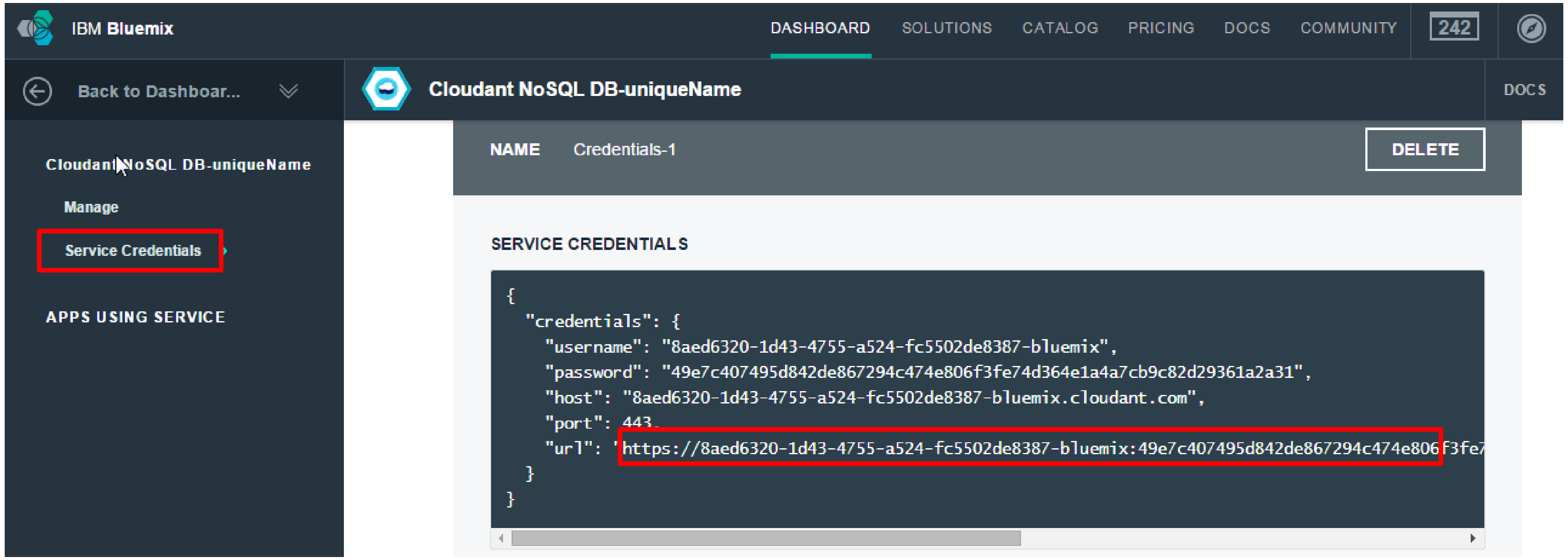
Create a database

1. Click **Add New Database**.
2. Enter `cars`.
3. Click **Create**.



Upload data to Cloudant database

1. Navigate to Cloudant DB service instance dashboard.
2. On the left navigation, click **Service Credentials**.
3. Copy the URL in the right main window to access the database remotely.
4. The URL format is `https://`*`username:password`*`@username.cloudant.com`



NAME Credentials-1

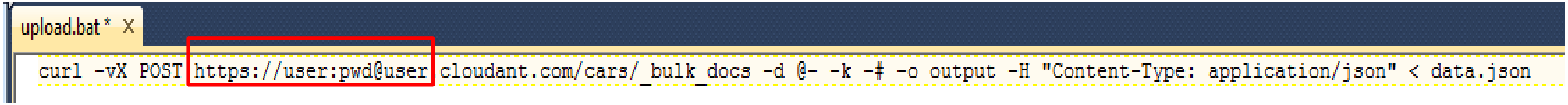
SERVICE CREDENTIALS

```
{
  "credentials": {
    "username": "8aed6320-1d43-4755-a524-fc5502de8387-bluemix",
    "password": "49e7c407495d842de867294c474e806f3fe74d364e1a4a7cb9c82d29361a2a31",
    "host": "8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com",
    "port": 443,
    "url": "https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix:49e7c407495d842de867294c474e806f3fe7
  }
}
```

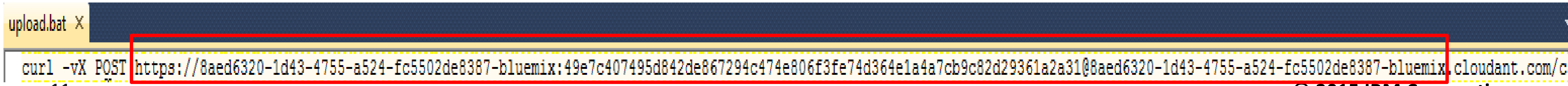
Upload data to Cloudant database, continued

4. Navigate to the local folder where you extracted the lab material from GitHub.
5. Open the upload.bat file so that you can edit it.
6. Replace the text in the red box with your Cloudant DB instance URL that you copied from the previous step.

The cURL tool enables data transfer with URL syntax in command lines or scripts. In this lab, the upload.bat file establishes a connection to your Cloudant database instance, uploads the initial JSON data documents contained in the data.json file, and records the server response in the output file.



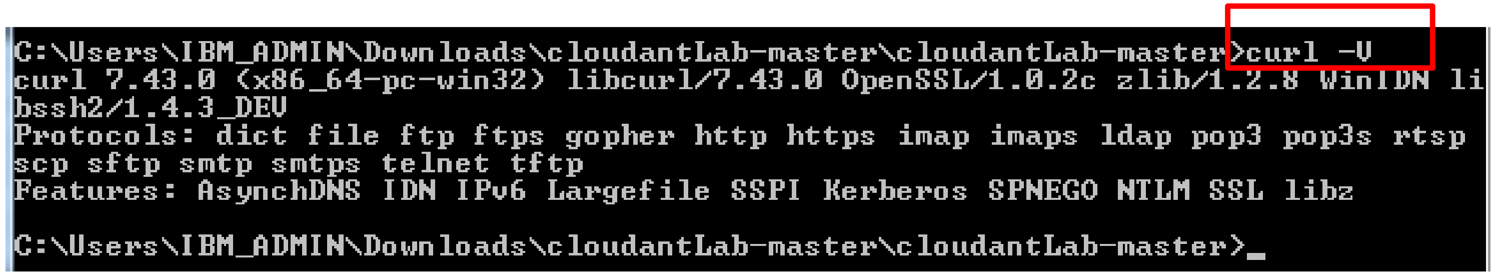
```
curl -vX POST https://user:pwd@user.cloudant.com/cars/_bulk_docs -d @- -k -# -o output -H "Content-Type: application/json" < data.json
```



```
curl -vX POST https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix:49e7c407495d842de867294c474e806f3fe74d364e1a4a7cb9c82d29361a2a31@8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/c
```

Upload data to Cloudant database, continued

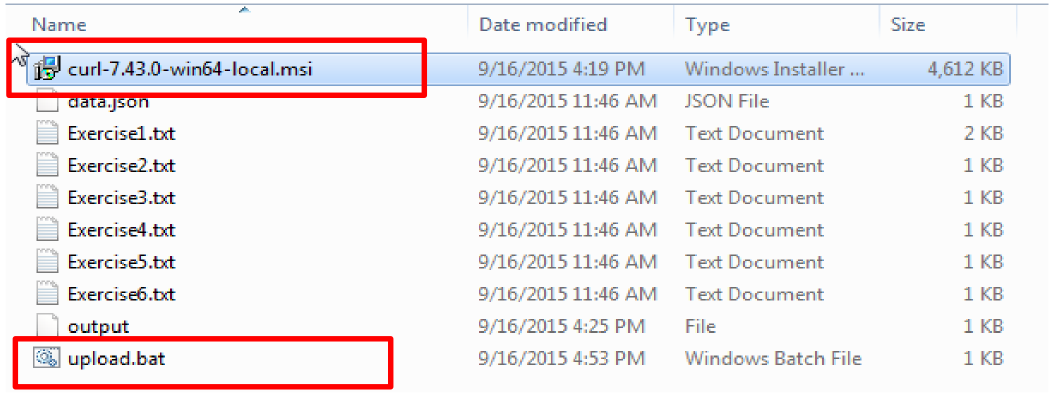
7. Double-click the upload.bat file. If it fails, ensure that your Cloudant database service instance URL included in the upload.bat file is correct.
8. Run the `curl -V` command. You should see the following output :



```
C:\Users\IBM_ADMIN\Downloads\cloudantLab-master\cloudantLab-master>curl -V
curl 7.43.0 (x86_64-pc-win32) libcurl/7.43.0 OpenSSL/1.0.2c zlib/1.2.8 WinIDN li
bssh2/1.4.3_DEV
Protocols: dict file ftp ftps gopher http https imap imaps ldap pop3 pop3s rtsp
scp sftp smtp smtps telnet tftp
Features: AsynchDNS IDN IPv6 Largefile SSPI Kerberos SPNEGO NTLM SSL libz

C:\Users\IBM_ADMIN\Downloads\cloudantLab-master\cloudantLab-master>_
```

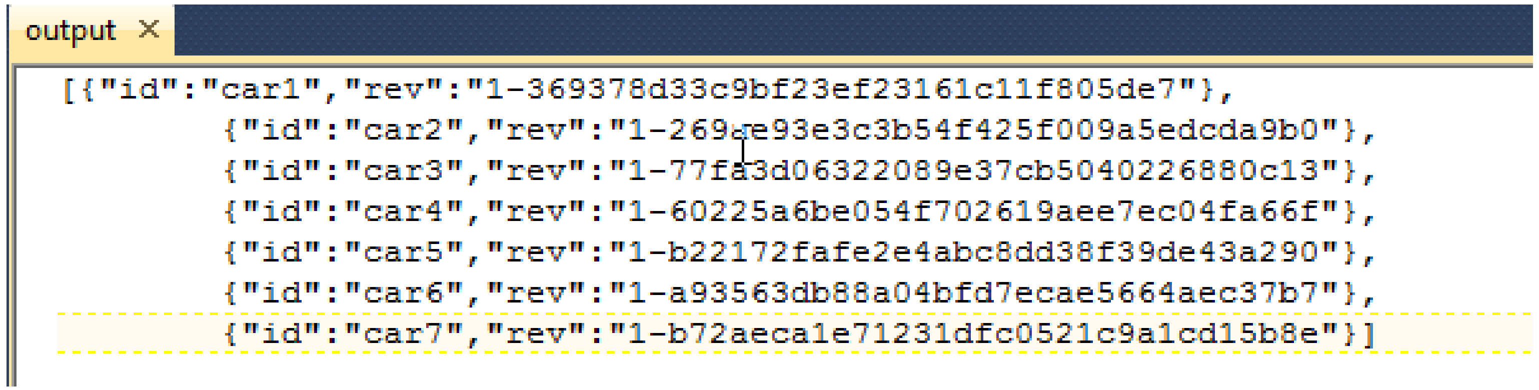
If the **curl** command is not recognized, install cURL using the curl-7.43.0-win64-local.msi file in the same extracted folder. Next, execute the upload.bat file.



Name	Date modified	Type	Size
curl-7.43.0-win64-local.msi	9/16/2015 4:19 PM	Windows Installer ...	4,612 KB
data.json	9/16/2015 11:46 AM	JSON File	1 KB
Exercise1.txt	9/16/2015 11:46 AM	Text Document	2 KB
Exercise2.txt	9/16/2015 11:46 AM	Text Document	1 KB
Exercise3.txt	9/16/2015 11:46 AM	Text Document	1 KB
Exercise4.txt	9/16/2015 11:46 AM	Text Document	1 KB
Exercise5.txt	9/16/2015 11:46 AM	Text Document	1 KB
Exercise6.txt	9/16/2015 11:46 AM	Text Document	1 KB
output	9/16/2015 4:25 PM	File	1 KB
upload.bat	9/16/2015 4:53 PM	Windows Batch File	1 KB

Upload data to Cloudant database, continued

8. Check the output file. If the BAT file ran successfully, you should see output similar to this:



```
[{"id":"car1","rev":"1-369378d33c9bf23ef23161c11f805de7"},
        {"id":"car2","rev":"1-269ae93e3c3b54f425f009a5edcda9b0"},
        {"id":"car3","rev":"1-77fa3d06322089e37cb5040226880c13"},
        {"id":"car4","rev":"1-60225a6be054f702619aee7ec04fa66f"},
        {"id":"car5","rev":"1-b22172fafe2e4abc8dd38f39de43a290"},
        {"id":"car6","rev":"1-a93563db88a04bfd7ecae5664aec37b7"},
        {"id":"car7","rev":"1-b72aeca1e71231dfc0521c9a1cd15b8e"}]
```

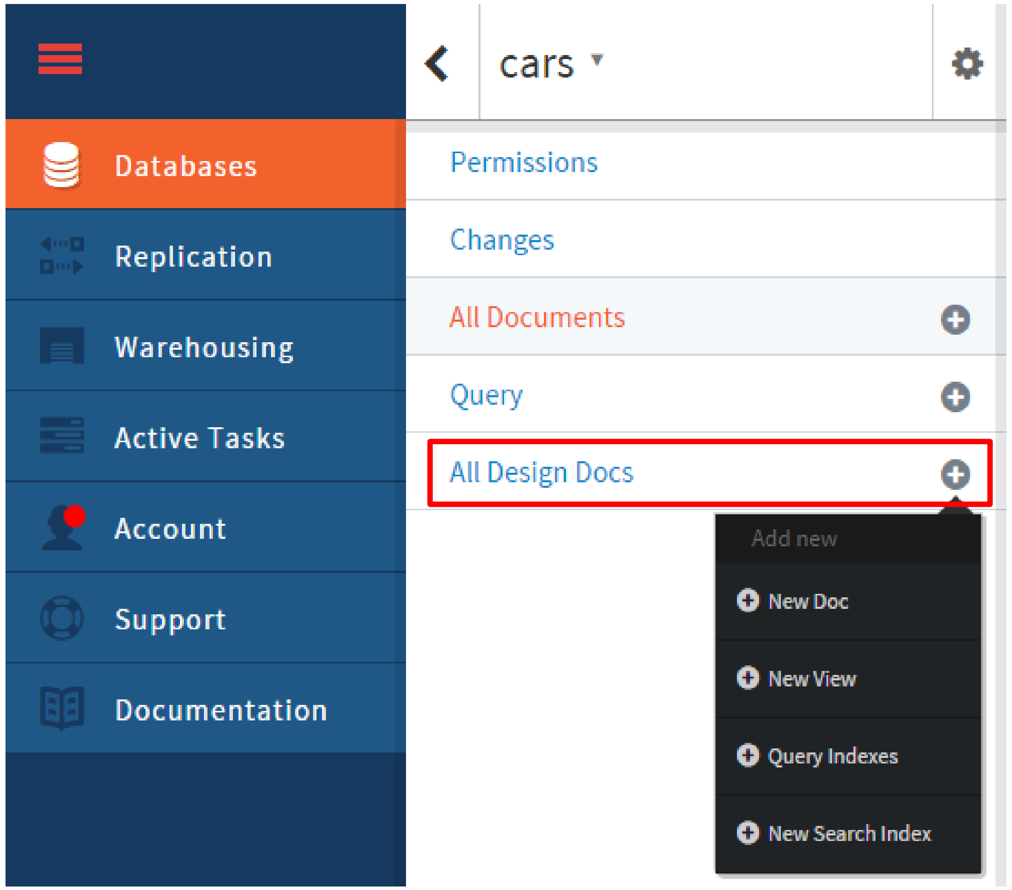
IBM developerWorks

Create views

Create a secondary index

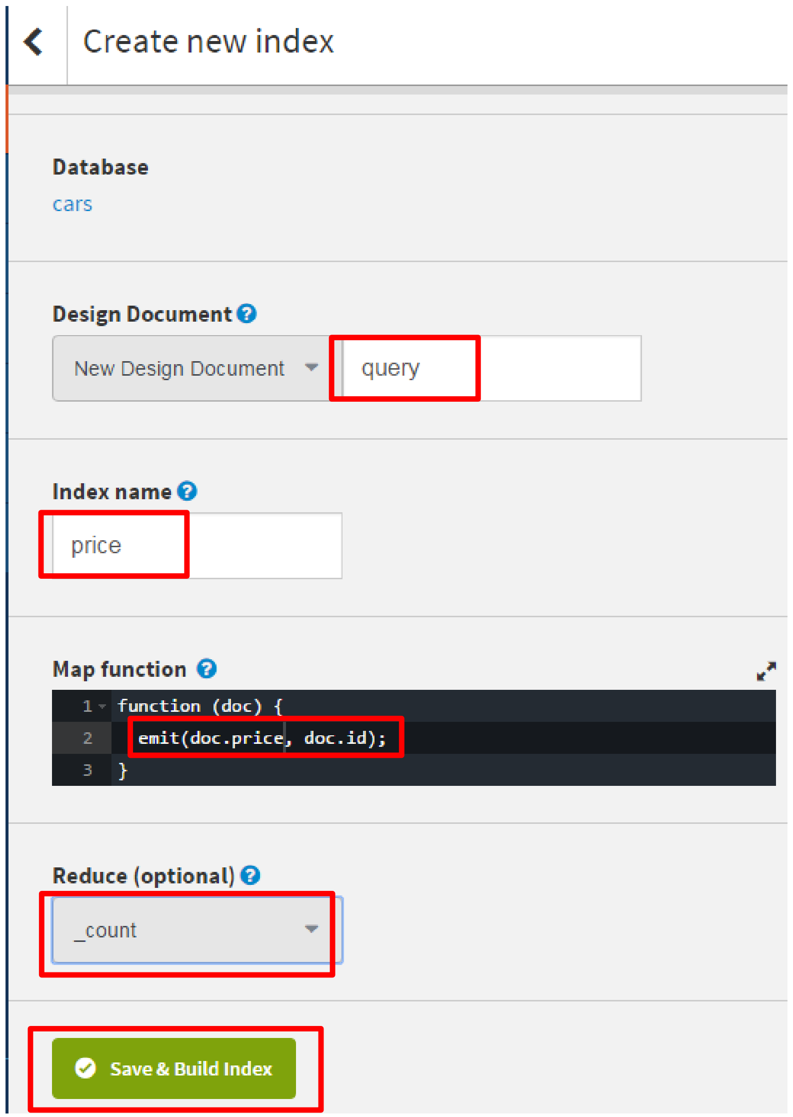
Creating a secondary index is ideal for routine queries. MapReduce is used to build an index for a large amount of data. These functions are written in JavaScript and held in design documents. For more information about secondary indexes, see https://cloudant.com/for-developers/views.

1. Navigate to the **cars** database.
2. Click the Plus sign (+) next to **All Design Docs**.
3. Click **New View**.



Create a secondary index, continued

4. Name the New Design Document, `query`.
5. Specify `price` as the Index name.
6. Modify the Map function by adding `emit(doc.price, doc._id);`
7. Select **_count** as the Reduce function.
8. Click **Save & Build Index**.


Create new index

Database
cars

Design Document
New Design Document | query

Index name
price

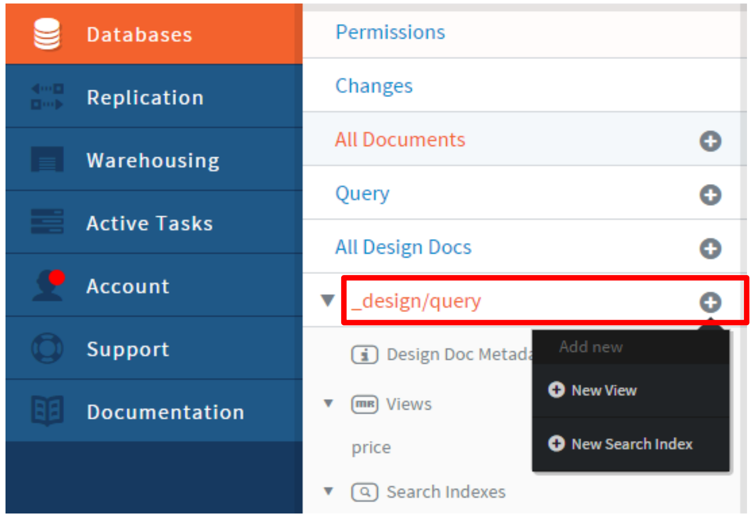
Map function

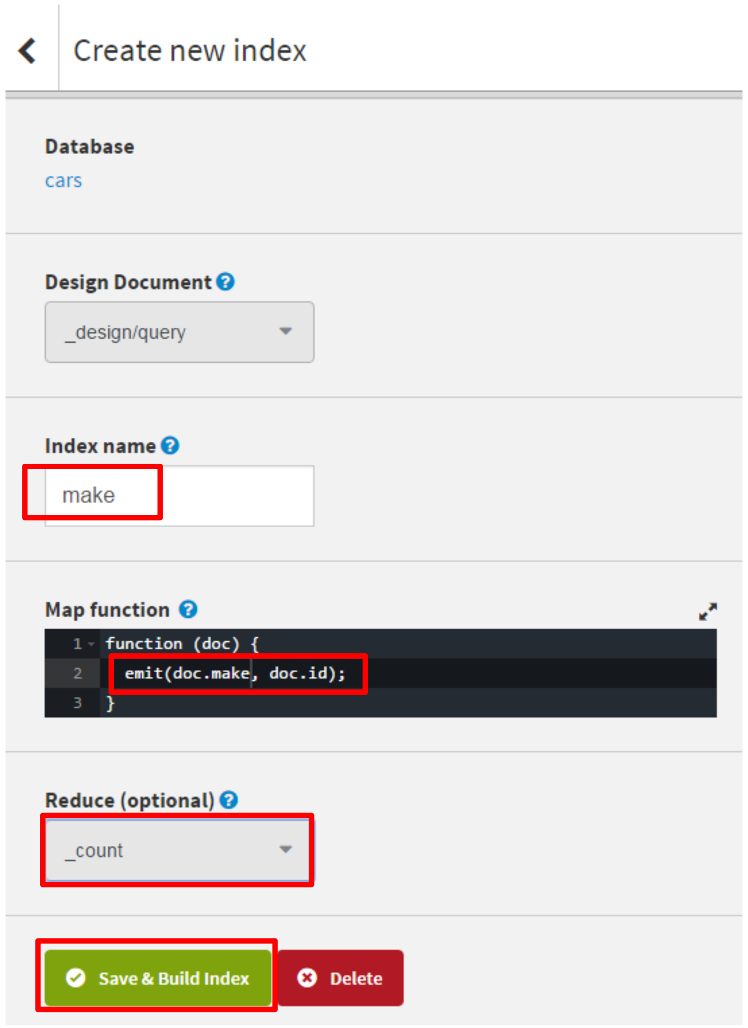
```
function (doc) {
  emit(doc.price, doc.id);
}
```

Reduce (optional)
_count

Create another secondary index

1. Click plus sign (+) next to **_design/query** that you created in previous step.
2. Click **New View**.
3. Specify `make` as the Index name.
4. Modify the Map function by adding `emit(doc.make, doc._id);`
5. Select **_count** as the Reduce function.
6. Click **Save & Build Index**.





Database
cars

Design Document
_design/query

Index name
make

Map function
```
function (doc) {
  emit(doc.make, doc.id);
}
```

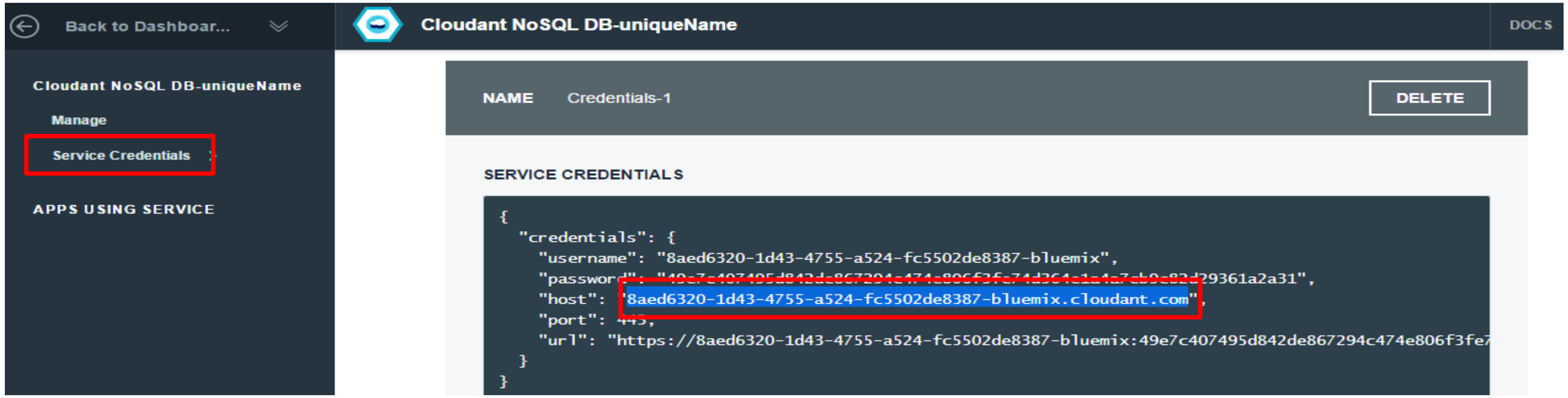
Reduce (optional)
_count

Save & Build Index

Make RESTful call to Cloudant using the Price view

The call to the Cloudant database that uses the Price view will return a unique price (key) and aggregated count (value) that is defined in the MapReduce function.

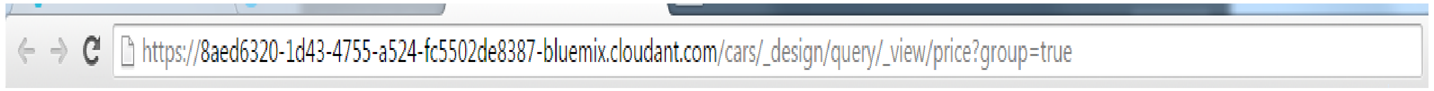
1. Get the Cloudant database URL by navigating to the Cloudant DB service instance on the main page.
2. In the left navigation, select **Service Credentials**.
3. Copy the host URL.



```
{
  "credentials": {
    "username": "8aed6320-1d43-4755-a524-fc5502de8387-bluemix",
    "password": "49e7c407495d842de867294c474e806f3fe74d364c1a4a7cb9c82d29361a2a31",
    "host": "8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com",
    "port": 443,
    "url": "https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix:49e7c407495d842de867294c474e806f3fe7
  }
}
```

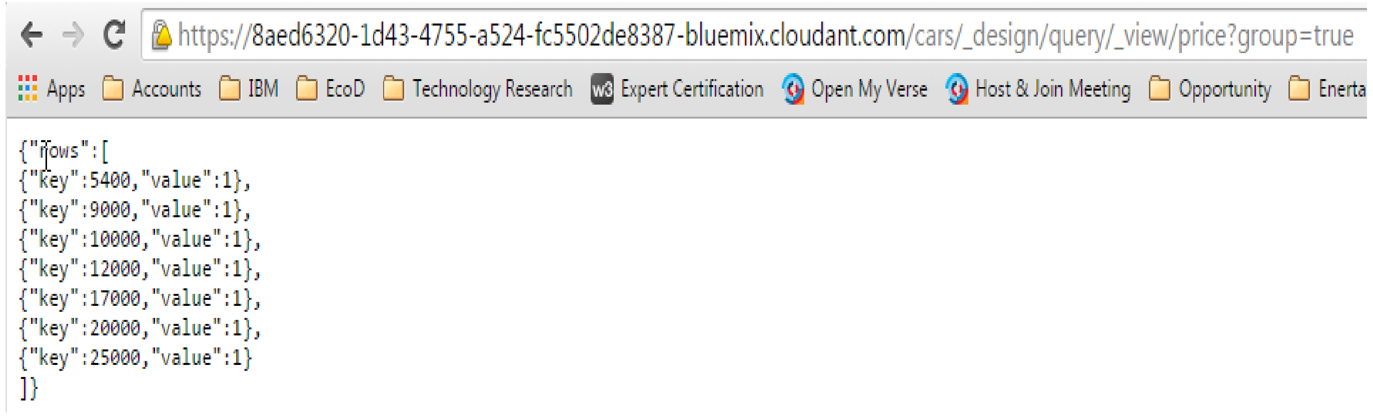
Make a RESTful call to Cloudant using the Price view, continued

4. In a browser, enter `https://` and the rest of the database host URL.
5. Enter `/cars/_design/query/_view/price?group=true`
6. Press Enter. If requested, enter the user name and password for the Service Credentials.



https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_view/price?group=true

You should receive a response similar to this in your browser.



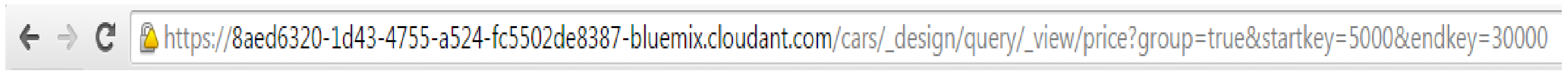
https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_view/price?group=true

```
{"rows":[
{"key":5400,"value":1},
{"key":9000,"value":1},
{"key":10000,"value":1},
{"key":12000,"value":1},
{"key":17000,"value":1},
{"key":20000,"value":1},
{"key":25000,"value":1}
]}
```

Make a RESTful call to Cloudant using the Price view for a price range

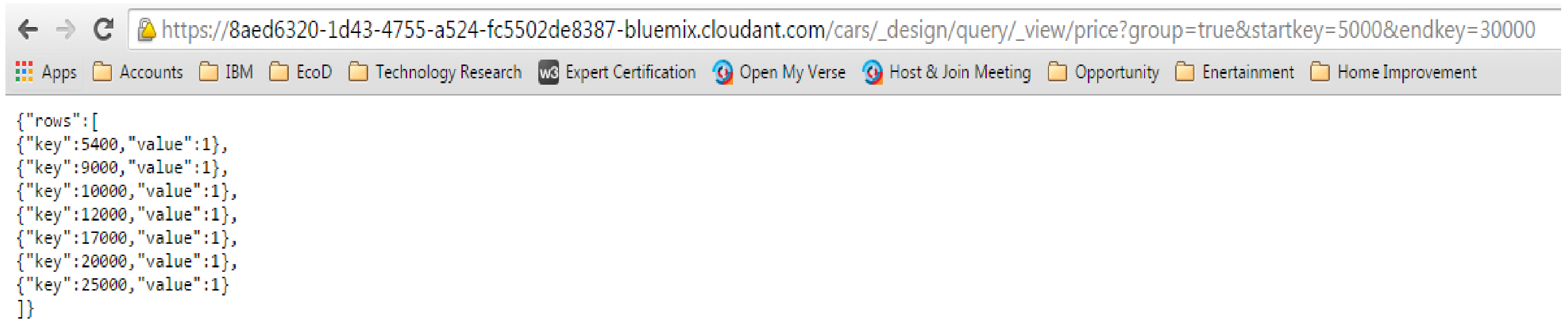
The range of prices are defined in the URL. Make a note on the usage of the startkey and the endkey with the price view.

1. In a browser, enter `https://` and the rest of the database host URL.
2. Enter `/cars/_design/query/_view/price?group=true&startkey=5000&endkey=30000`
3. Press Enter.



https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_view/price?group=true&startkey=5000&endkey=30000

You should receive a response similar to this.

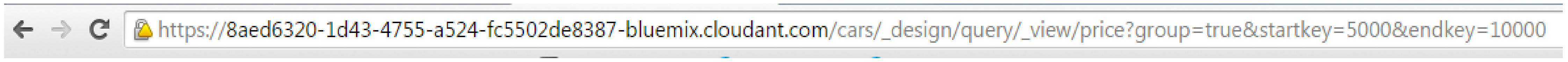


https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_view/price?group=true&startkey=5000&endkey=30000

```
{"rows":[
{"key":5400,"value":1},
{"key":9000,"value":1},
{"key":10000,"value":1},
{"key":12000,"value":1},
{"key":17000,"value":1},
{"key":20000,"value":1},
{"key":25000,"value":1}
]}
```

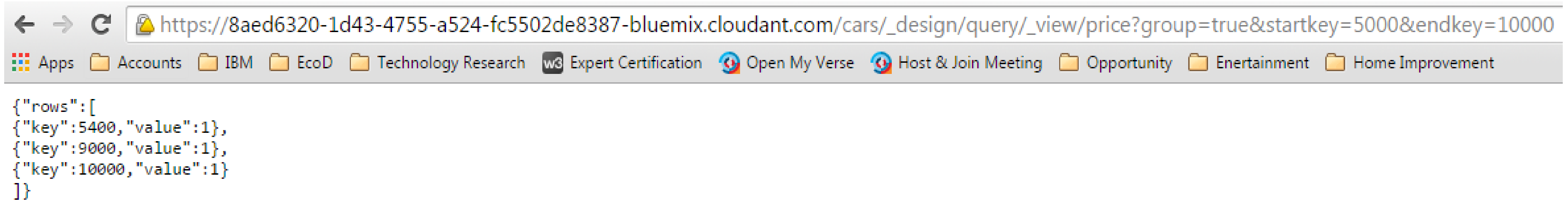
Make a RESTful call to Cloudant using the Price view for a price range with an upper limit

4. Change the range to see certain return values. In the REST call URL, specify the endkey value as `10000`.
5. Press Enter.



```
https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_view/price?group=true&startkey=5000&endkey=10000
```

The price (unique key) and aggregated count (value) that is over 10,000 is not returned in response



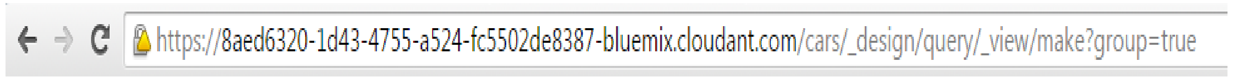
```
https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_view/price?group=true&startkey=5000&endkey=10000
```

```
{"rows":[
{"key":5400,"value":1},
{"key":9000,"value":1},
{"key":10000,"value":1}
]}
```

Make a RESTful call to Cloudant using the Make view

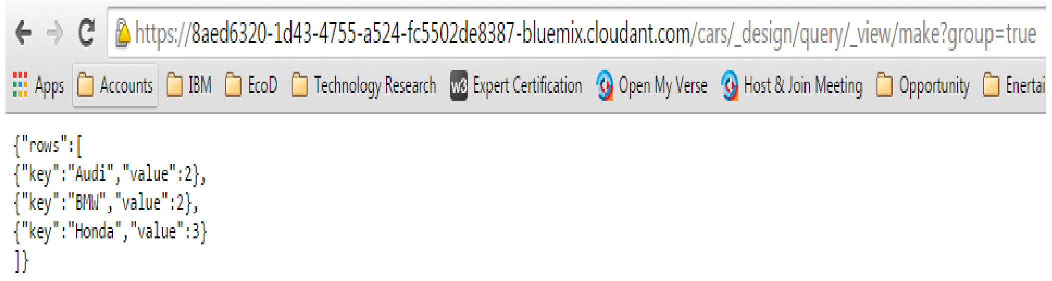
The call to the Cloudant database that uses Make view will return a unique make (key) and an aggregated count (value) that is defined in the MapReduce function.

1. In a browser, enter `https://` and rest of the database host URL.
2. Enter `/cars/_design/query/_view/make?group=true`
3. Press Enter.



https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_view/make?group=true

You should receive a response similar to this.



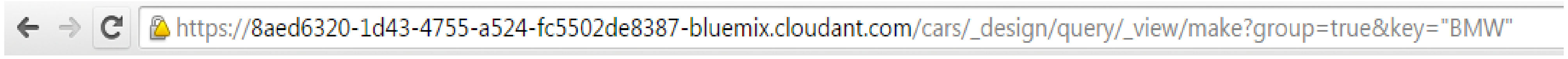
https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_view/make?group=true

```
{"rows":[
{"key":"Audi","value":2},
{"key":"BMW","value":2},
{"key":"Honda","value":3}
]}
```

Make RESTful call to Cloudant using the Make view for a specific model search

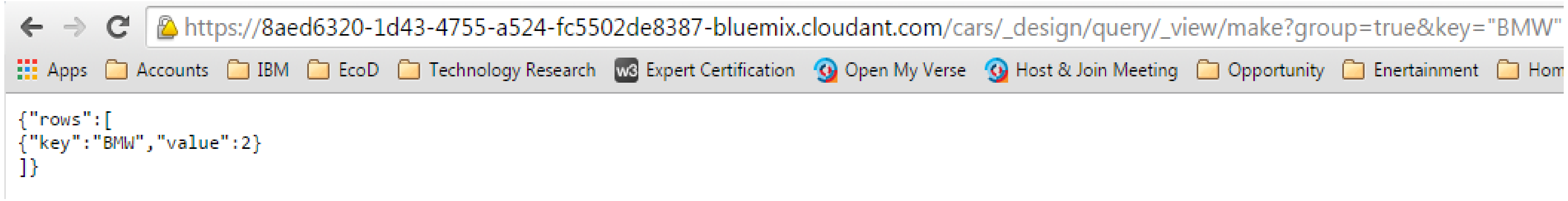
The call to the Cloudant database that uses the Make view targets a specific model that is indicated by the key parameter in the URL.

1. In a browser, enter `https://` and the rest of the database host URL.
2. Enter `/cars/_design/query/_view/make?group=true&key=` “BMW”
3. Press Enter.



https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_view/make?group=true&key="BMW"

You should receive a response similar to this.



https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_view/make?group=true&key="BMW"

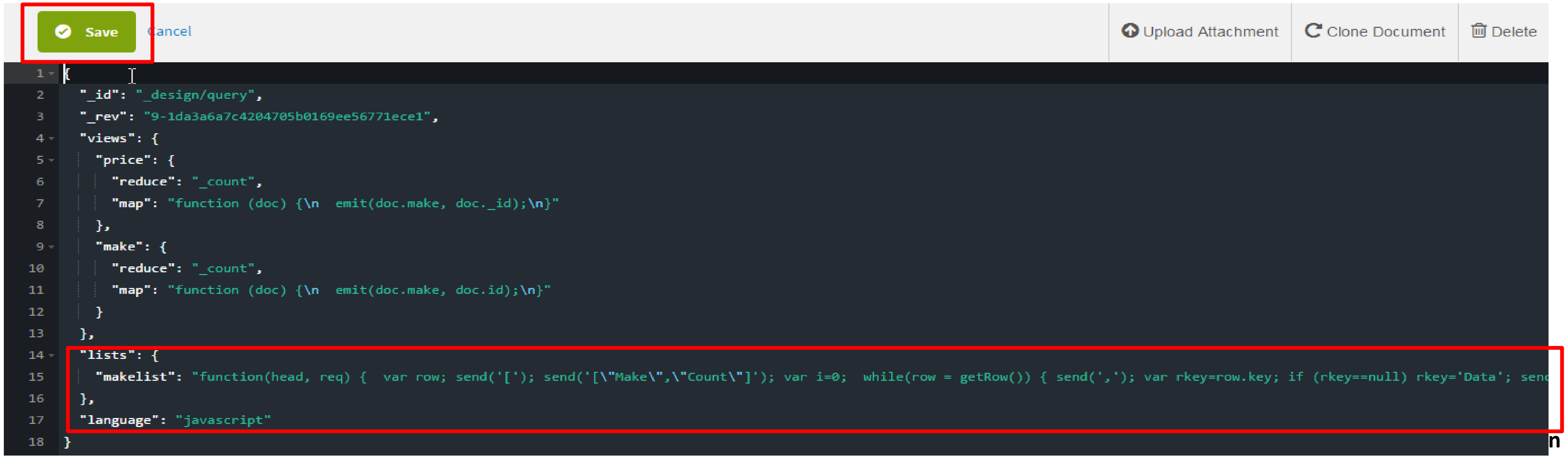
```
{"rows":[
{"key":"BMW","value":2}
]}
```

Make a RESTful call to Cloudant using the List function

1. Add lists by inserting the following code in the Makelist function:

Note: Place the semi colon(,) prior to inserting the following code

```
"lists": {
    "makelist": "function(head, req) {  var row; send('['); send('[\"Make\",\"Count\"]'); var i=0;
while(row = getRow()) { send(','); var rkey=row.key; if (rkey==null) rkey='Data';
send('[\"'+rkey+'\",'+row.value+']'); i++; }send(']');}"
  },
  "language": "javascript"
```



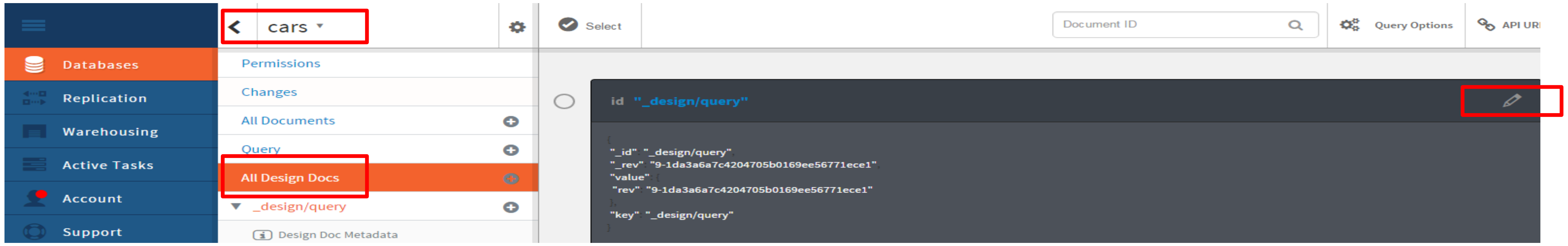
Save Cancel Upload Attachment Clone Document Delete

```
{
  "_id": "_design/query",
  "_rev": "9-1da3a6a7c4204705b0169ee56771ece1",
  "views": {
    "price": {
      "reduce": "_count",
      "map": "function (doc) {\n  emit(doc.make, doc._id);\n}"
    },
    "make": {
      "reduce": "_count",
      "map": "function (doc) {\n  emit(doc.make, doc.id);\n}"
    }
  },
  "lists": {
    "makelist": "function(head, req) {  var row; send('['); send('[\"Make\",\"Count\"]'); var i=0;  while(row = getRow()) { send(','); var rkey=row.key; if (rkey==null) rkey='Data'; send
  },
  "language": "javascript"
}
```

Make a RESTful call to Cloudant using the List function, continued

With the List function, you can customize the format of MapReduce query results. You will add a list function that is written in JavaScript to return data in [Make, Count] list format from the query result by using the Make view.

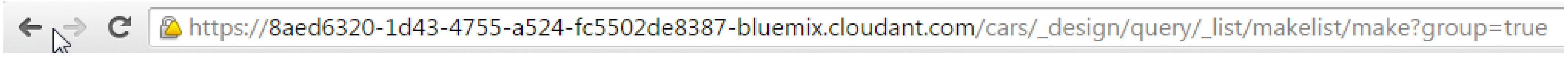
1. Navigate to Cloudant database management view.
2. Select cars database and click **All Design Docs**.
3. Click the Pencil icon to edit the query design document.



```
{
"_id": "_design/query",
"_rev": "9-1da3a6a7c4204705b0169ee56771ece1",
"value": {
 "rev": "9-1da3a6a7c4204705b0169ee56771ece1"
},
"key": "_design/query"
}
```

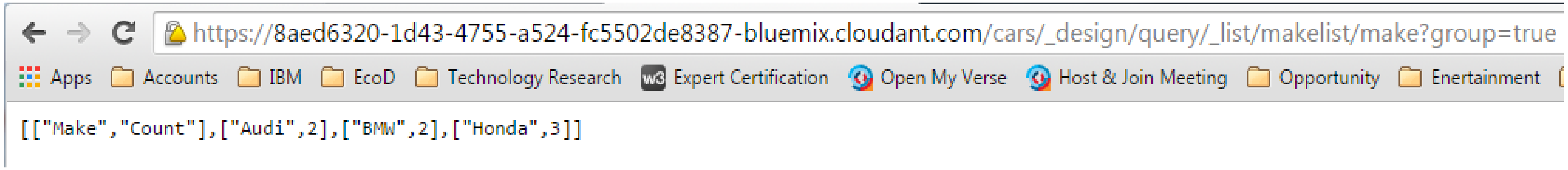
Test the List function

1. In a browser, enter `https://` and the rest of the database host URL.
2. Enter `/cars/_design/query/_list/makelist/make?group=true`
3. Press Enter.



https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_list/makelist/make?group=true

Data returns in list format with Make and Count as defined in the query design.



https://8aed6320-1d43-4755-a524-fc5502de8387-bluemix.cloudant.com/cars/_design/query/_list/makelist/make?group=true

```
[["Make","Count"],["Audi",2],["BMW",2],["Honda",3]]
```

Recap

- In this session, you learned how to:
  - Create an unbound Cloudant SQL database instance
  - Use cURL as part of upload.bat file to establish a connection between your local file and the Cloudant instance to transfer data
  - Create views by using MapReduce to build secondary index, which is ideal for routine queries
  - Use REST calls to access data from Cloudant database
  - Use the List function to customize the format of MapReduce query results from REST calls