**CAPM Documentation for Beginner**

1. Links: [Querying in JavaScript | capire (cloud.sap)](https://cap.cloud.sap/docs/node.js/cds-ql)
2. OData V4 Doc : [Model Instantiation and Data Access | SAP Help Portal](https://help.sap.com/docs/SAP_NETWEAVER_AS_ABAP_752/468a97775123488ab3345a0c48cadd8f/9613f1f2d88747cab21896f7216afdac.html)
3. : [Step 1: The Initial App - Documentation - Demo Kit - SAPUI5 SDK (ondemand.com)](https://sapui5.hana.ondemand.com/sdk/#/topic/15d84f36c0594cb2b3295aa1f55cb961)

1. To create a create a table in sqlite3 and retrieve data.

* Go to schema.cds in db folder and add following:

namespace my.bookshop;

using { Country, managed } from '@sap/cds/common';

entity Books {

  key ID : Integer;

  title  : localized String;

  author : localized String;

  stock  : Integer;

}

The above will define the structure of the table Books. Bookshop is the dbname is sqlite.

* The above will create a my.bookshop-Books.csv file inside db>data folder by saving and typing cds add data command in the terminal, then populate the table with the following data.

ID,title,author\_ID,stock

201,Wuthering Heights,101,12

207,Jane Eyre,107,11

251,The Raven,150,333

252,Eleonora,150,555

271,Catweazle,170,22

* Now go to the cat-service.cds file inside srv folder to and add the following,

using my.bookshop as my from '../db/schema';

service CatalogService {

  entity Books @readonly as projection on my.Books;

}

* Then go to the terminal and type cds deploy to insert the new data into your sqlite db and use cds watch to run the application.

2. Test the odata via REST API

* Create a test.http file in the root folder and add the following,

###

#

# Browse Books

#

GET http://localhost:4004/odata/v4/catalog/Books?

  # &$select=title,stock

###

#

# Browse book by id

GET http://localhost:4004/odata/v4/catalog/Books(201)

3. Curd Operations

* Change Permissions by changing   
  entity Books @readonly as projection on my.Books;

To

entity Books as projection on my.Books;

* add the following in cat-service.js

  // update

// For update, use UPDATE instead of CREATE,

// https://cap.cloud.sap/docs/guides/providing-services#custom-logic

  srv.before ('CREATE', 'Orders', async (req) => {

    const order = req.data

    if (!order.amount || order.amount <= 0)  return req.error (400, 'Order at least 1 book')

    const tx = cds.transaction(req)

    const affectedRows = await tx.run (

      UPDATE (Books)

        .set   ({ stock: {'-=': order.amount}})

        .where ({ stock: {'>=': order.amount},/\*and\*/ ID: order.book\_ID})

    )

    if (affectedRows === 0)  req.error (409, "Sold out, sorry")

  })

  //delete

  srv.on('DELETE', 'Books', async (req) => {

    const book\_ID = req.data.ID

    if (book\_ID === '') req.error(405, "Please enter a book id")

    const tx = cds.transaction(req)

    const affectedRows = await tx.run(

      DELETE.from(Books)

        .where({ ID: {'=':book\_ID} })

    )

    if (affectedRows === 0) req.error(404, "Book not found")

  })

  //insert

  srv.before('CREATE', 'Orders', async (req) => {

    const newBook = req.data

    if (!newBook.ID || !newBook.title || !newBook.author\_ID || !newBook.stock) {

      return req.error(400, 'All book details must be provided')

    }

    const tx = cds.transaction(req)

    await tx.run(

      INSERT.into(Books).entries(newBook)

    )

  })

  // Add some discount for overstocked books

  srv.after ('READ', 'Books', each => {

    if (each.stock > 111)  each.title += ' -- 11% discount!'

  })

* Add the following in test.http

### Update a Book

POST http://localhost:4004/odata/v4/catalog/Orders

Content-Type: application/json;IEEE754Compatible=true

{"book\_ID": 271, "amount": 2}

### Insert a new book

POST http://localhost:33807/odata/v4/catalog/Books

Content-Type: application/json

{

  "ID": 301,

  "title": "cspsyco2",

  "author\_ID": 170,

  "stock": 50

}

### Delete a book

DELETE http://localhost:33807/odata/v4/catalog/Books(301)

# Browse book by id

GET http://localhost:33807/odata/v4/catalog/Books(201)

* All the above code added cat-service.js is custom logic, those are not necessary to do curd operations, you can remove it and send request, still it works apart from the update operation. For updating the data use the follow code in test.http file.

###update without custom logic

PUT http://localhost:4004/odata/v4/catalog/Books(101)

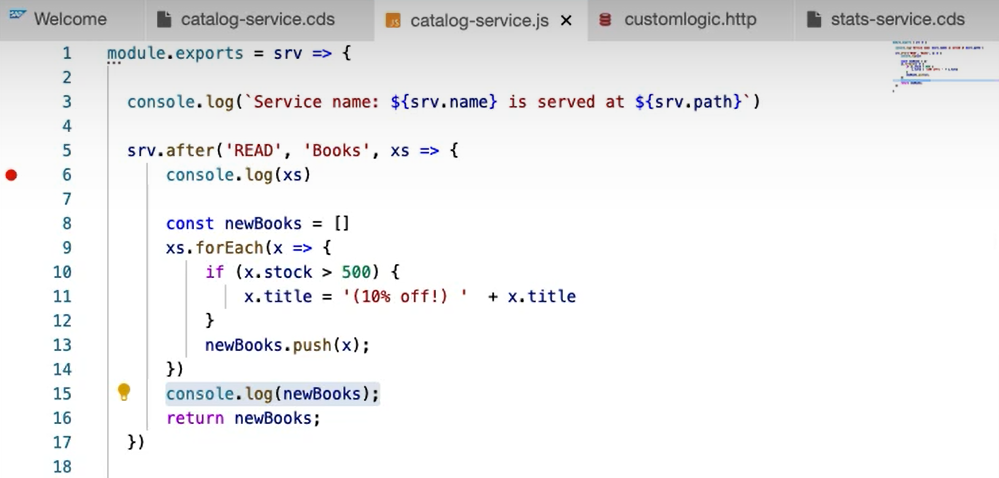
Content-Type: application/json

{"ID": 101, "title": "Some Author3"}

Reference: [Providing Services | capire (cloud.sap)](https://cap.cloud.sap/docs/guides/providing-services#custom-logic)

Note: Compilation command=>$ cds compile srv/stat-service.cds -2 sql

* Another simple Custom Read Logic is given below,



4. By default, it is using in memory database which is persistent. To connect this app to a sqlite database, add the following in package.json.  
  
  "cds": {

    "requires": {

      "db": {

        "kind": "sqlite",

        "credentials": {

          "url": "db/my-bookshop.sqlite"

        }

      }

    }

  }

5. IDK: Persist Data by Deploying to SQLite with cds deploy --to sqlite

Instead of using in-memory, we can also use persistent databases, e.g. SQLite. The difference from the automatically provided in-memory database is that we now get a persistent database stored in the local file ./sqlite.db. This is also recorded in the package.json.

cds deploy --to sqlite

Then we can use the sqlite3 CLI to query with the newly created database:

sqlite3 sqlite.db .dump

sqlite3 sqlite.db .tables

6. To update the schema of a table without losing the data in the table, add the following line in package.json

"cds": {

    "requires": {

      "db": {

        "kind": "sqlite",

        "credentials": {

          "url": "db/my-bookshop.sqlite"

        },

        "schema\_evolution": "auto"

      }

    }

  },

Note: You can only do in the beginning before using cds deploy. If you have already done the deployment, you don’t have any other option other than deleting the whole database and redeploy. You can do it by the following steps.  
  
a. Go to db folder and delete *your-database.sqlite.*

b. Add schema evolution to package.json.

c. Use cds deploy command to create the new database.

d. Now you can change the db schema and use cds deploy command update the table.

e. Reference: <https://cap.cloud.sap/docs/guides/databases-sqlite>

**REFERENCES:**

Part 1: ODATA CREATION

[Create a CAP Business Service with Node.js Using Visual Studio Code | SAP Tutorials](https://developers.sap.com/tutorials/cp-apm-nodejs-create-service.html#b8c14c60-e252-4cd2-8e62-86d2329f564d)

Part 2: ODATA CURD OPERATIONS COMPLETE COURSE

<https://youtube.com/playlist?list=PL6RpkC85SLQBFi4SK77b2y4EwlXMVG0XJ&si=c6LD9ypP_n57FtND>