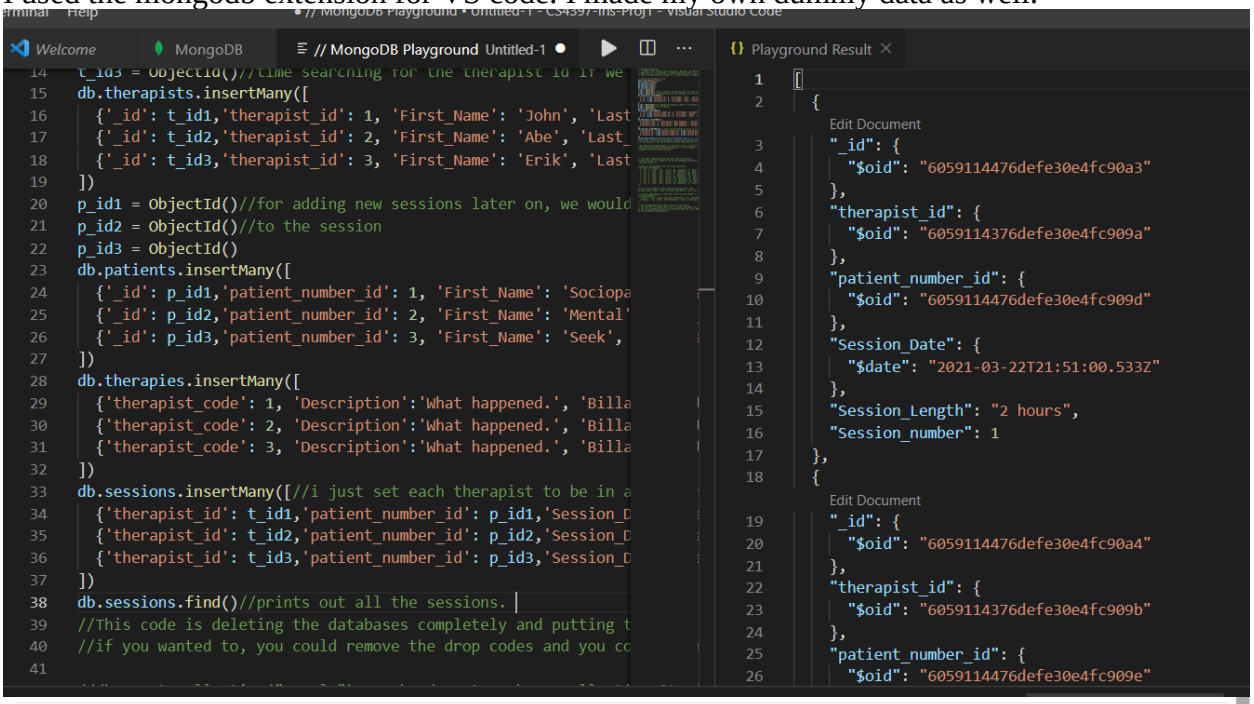
Dominic Grant

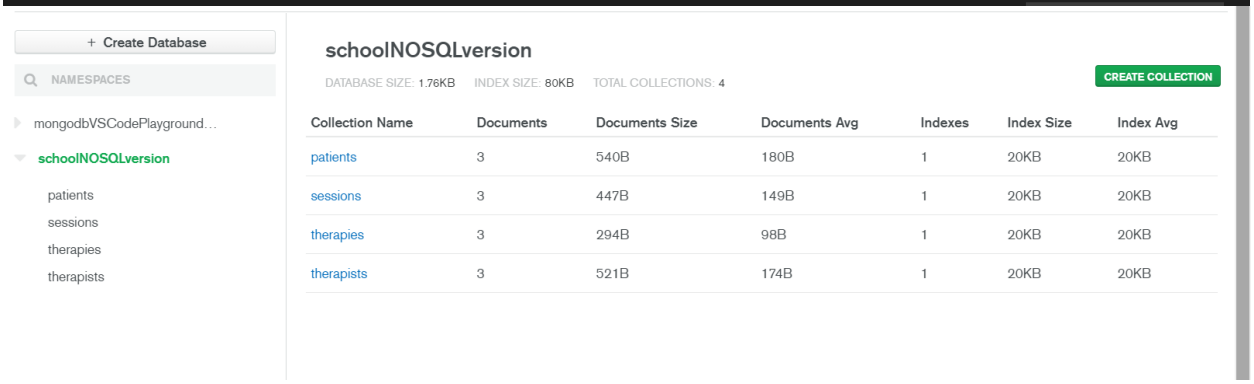
Final submission for SQL/NoSQL (MongoDB) Project:

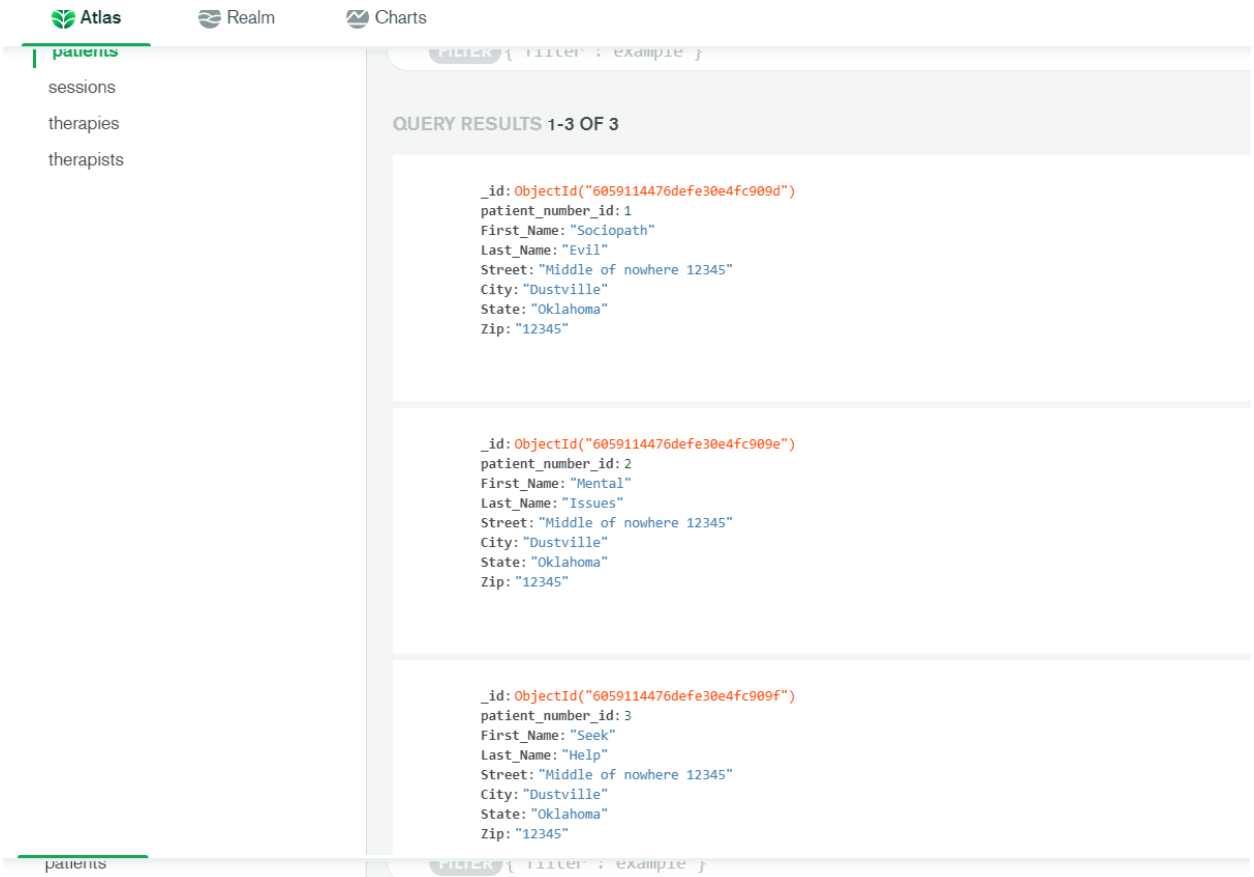
The SQL portion I chose from old projects.

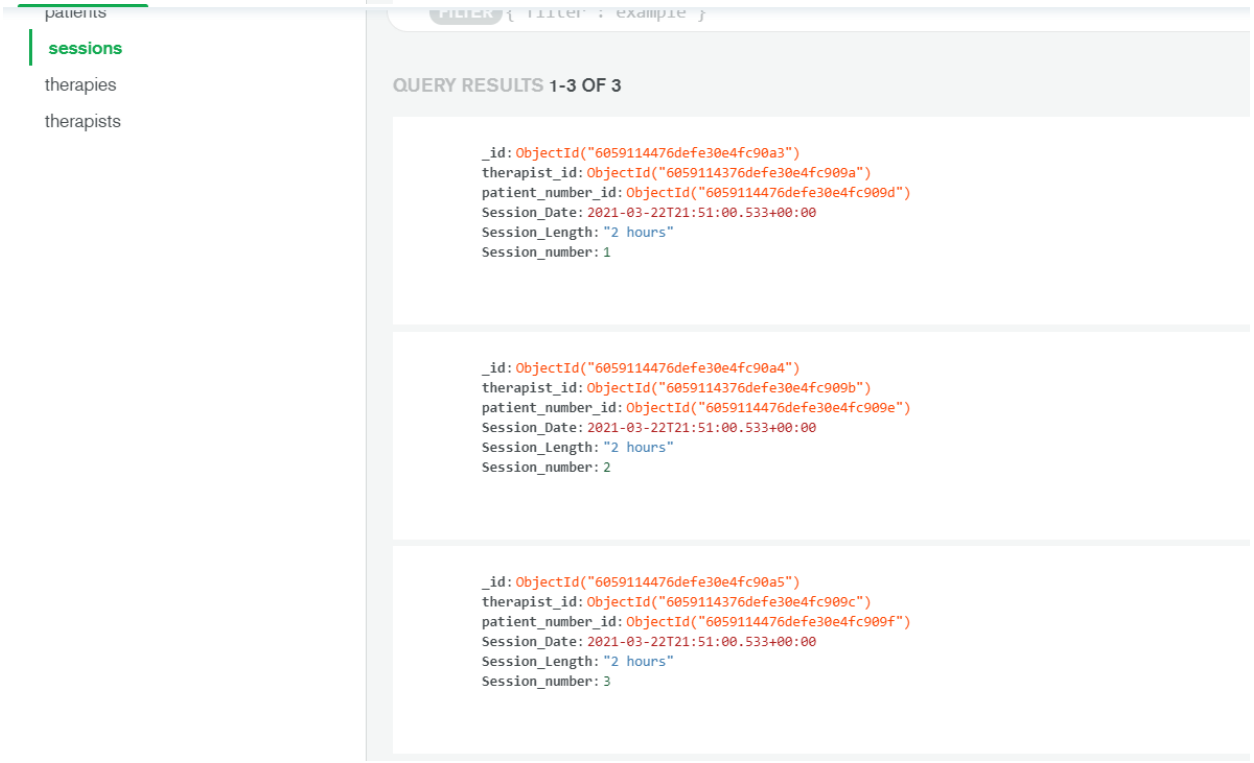
```
1.Table Patients(Patient_number INTEGER,
                 First_name char(30),
                 Last_name char(30),
                     Address char(30),
                 City char(30),
                 State char(30),
                 Zip char(5),
                 Balance INTEGER,
                 PRIMARYKEY(Patient_number));
2.Table Therapists(therapist_ID INTEGER,
                 First_name char(30),
                 Last_name char(30),
                     Street char(30),
                 City char(30),
                 State char(30),
                 Zip char(5),
                 PRIMARYKEY(therapist_ID));
3.Table Therapies(therapist_code INTEGER,
                 description char(30),
                 billable_unit char(30),
                     PRIMARYKEY(therapist_code));
4.Table session(therapist_id INTEGER,
                 Session_date DATE,
                 Session_length char(30),
                     Session_number INTEGER,
                 FOREIGNKEY(Therapist_id) REFERENCE therapist(therapist_id),
          FOREIGNKEY(Patient_number) references to patients(patient_number);
```

I used the mongodb extension for VS code. I made my own dummy data as well.






Atlas
Realm
Charts
patients
sessions
therapies
therapists
{ filter : example }
QUERY RESULTS 1-3 OF 3
_id: ObjectId("6059114476defe30e4fc909d")
patient_number_id: 1
First_Name: "Sociopath"
Last_Name: "Evil"
Street: "Middle of nowhere 12345"
City: "Dustville"
State: "Oklahoma"
Zip: "12345"
_id: ObjectId("6059114476defe30e4fc909e")
patient_number_id: 2
First_Name: "Mental"
Last_Name: "Issues"
Street: "Middle of nowhere 12345"
City: "Dustville"
State: "Oklahoma"
Zip: "12345"
_id: ObjectId("6059114476defe30e4fc909f")
patient_number_id: 3
First_Name: "Seek"
Last_Name: "Help"
Street: "Middle of nowhere 12345"
City: "Dustville"
State: "Oklahoma"
Zip: "12345"


patients
sessions
therapies
therapists
{ filter : example }
QUERY RESULTS 1-3 OF 3
_id: ObjectId("6059114476defe30e4fc90a3")
therapist_id: ObjectId("6059114376defe30e4fc909a")
patient_number_id: ObjectId("6059114476defe30e4fc909d")
Session_Date: 2021-03-22T21:51:00.533+00:00
Session_Length: "2 hours"
Session_number: 1
_id: ObjectId("6059114476defe30e4fc90a4")
therapist_id: ObjectId("6059114376defe30e4fc909b")
patient_number_id: ObjectId("6059114476defe30e4fc909e")
Session_Date: 2021-03-22T21:51:00.533+00:00
Session_Length: "2 hours"
Session_number: 2
_id: ObjectId("6059114476defe30e4fc90a5")
therapist_id: ObjectId("6059114376defe30e4fc909c")
patient_number_id: ObjectId("6059114476defe30e4fc909f")
Session_Date: 2021-03-22T21:51:00.533+00:00
Session_Length: "2 hours"
Session_number: 3

patients

sessions

therapies

therapists

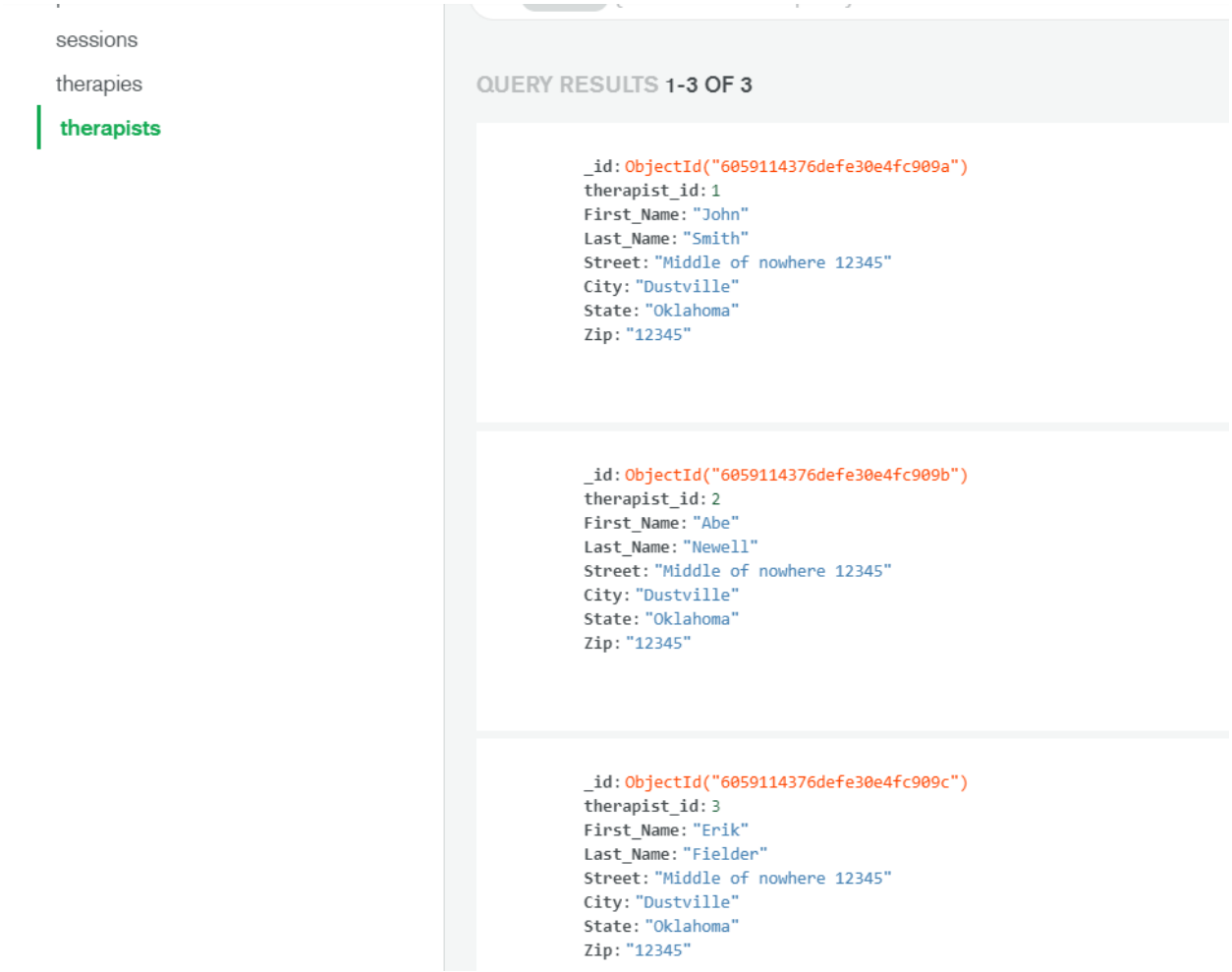
FILTER { filter : example }

QUERY RESULTS 1-3 OF 3

```
_id: ObjectId("6059114476defe30e4fc90a0")
therapist_code: 1
Description: "What happened."
Billable_Unit: "500$"
```

```
_id: ObjectId("6059114476defe30e4fc90a1")
therapist_code: 2
Description: "What happened."
Billable_Unit: "500$"
```

```
_id: ObjectId("6059114476defe30e4fc90a2")
therapist_code: 3
Description: "What happened."
Billable_Unit: "500$"
```



I research queries and the modifiers that both the SQL and NoSQL use. I made several different queries that do the same thing for both programs.

I first print out all 4 tables completely. Next, I print out sessions that have a therapist_id that's equal to 1. Then I print out therapists that have a Last_name that has a e in it. That's followed by a query that prints out patients that have a patient_number of 1 or 3. Then another query that prints out patients that have a Balance between 1000 and 1500. Then finally, one more query that prints out patients with their Balance in ascending order.

I did these in both programs making use of their various syntaxes. Here are them below.

SQL:

```
String query = "SELECT * FROM therapists";
query = "SELECT * FROM patients";
query = "SELECT * FROM therapies";
query = "SELECT * FROM sessions";
query = "SELECT * FROM sessions WHERE therapist_id = 1";
query = "SELECT * FROM therapists WHERE Last_name LIKE '%e%'";
query = "SELECT * FROM patients WHERE patient_number IN (1,3)";
query = "SELECT * FROM patients WHERE Balance BETWEEN 1000 AND 1500";
query = "SELECT * FROM patients ORDER BY Balance";
```

NoSQL (MongoDB):

```
const myCursor = db.therapists.find();
myCursor = db.patients.find();
myCursor = db.therapies.find();
myCursor = db.sessions.find();
myCursor = db.sessions.find({'therapist_id': t_id1});
myCursor = db.therapists.find({'Last_Name': { $regex: /e/}});
myCursor = db.patients.find({'patient_number_id': { $in: [1,3]}});
myCursor = db.patients.find({'Balance': { $gte: 1000, $lte: 1500}});
myCursor = db.patients.find().sort({'Balance': 1});
```

I put commented info in the code for some of the ones that aren't as straightforward and obvious in their use.

I also looked into triggers as asked. I got a simple trigger that adds in the therapies table once it's detected that the patients table has been inserted into for my SQL code.

```
why.execute("CREATE TRIGGER trig1 AFTER INSERT ON patients REFERENCING NEW AS upd
atedrow FOR EACH ROW MODE DB2SQL INSERT INTO therapies VALUES(updatedrow.patient_
number,'What happened.','500$')");
        System.out.println("Created trigger that does the therapies table after a
 new row is inserted into patients.");
```

Basically upon a insertion into the patient table it takes the patient_number from that inserted information, and puts that as the therapy number in the therapies insertion, along with the generic info and cost. More commented info in the code.

I did have a bug with implementing my MongoDB trigger however. MongoDB's triggers make use of serverside code that you customize to detect when something happens, and to trigger the trigger so to speak, so it's not like the SQL where you just put it in the same code.
My MongoDB trigger works when it has a test run done, and inputs the patient data, but when I run my MongoDB playground code, it acts like it doesn't see that. I have gotten in touch with MongoDB support to see if they can help, but as of now they haven't gotten back with info on why this is happening.

Operation Type

This trigger will only execute on these operations.

☑ Insert ☐ Update ☐ Delete ☐ Replace

Full Document

By turning on Full Document, you will receive the document created or modified in your change event. For Delete operations, the full document will not exist.

OFF

✓ FUNCTION

Select An Event Type

Learn how to use Amazon EventBridge with Triggers

Function | EventBridge

Function

```
exports = function(changeEvent) {
      const collection = context.services.get('Cluster0').db('schoolNOSQLversion').collection('therapies');
      collection.insertOne({therapist_code: 1, Description: 'What happened.', Billable_Unit: '500$'});
      collection.insertOne({therapist_code: 2, Description: 'What happened.', Billable_Unit: '500$'});
      collection.insertOne({therapist_code: 3, Description: 'What happened.', Billable_Unit: '500$'});
};
```