Unit 9: Databases & The Modern Web – Practice Assessment

Assigned: 03/06/23

Due: N/A

Completed: 03/06/23

**LMS:** <https://lms.grandcircus.co/mod/assign/view.php?id=22873>

**Google Doc:** <https://docs.google.com/document/d/1DIiuPJre_IcYJL27vqLUzBOenAmk4Hk4tC28ztZdxHI/preview>

**GitHub:** <https://github.com/gc-submissions/practice-assessment-8-clintmsmith>

**PART 1 - SQL**

**Overview:** Within PGAdmin, run the provided setup code to create the initial tables. Then complete SQL code for the four numbered bullets below.

**Setup:**

First, copy-paste the code from part1-setup.sql into PG Admin. (Use the Query Tool. After you run it, refresh to see the new tables.) It creates two tables: **boxes** and **reviews**. It also adds three rows to the **boxes** table and 5 rows to the **reviews** table. This provides the starting point for the questions below.

**Submission:**

In the provided SQL file part1-answers.sql, paste the SQL code for each of the four numbered bullets below. You do not need to submit the results of the queries, just the SQL code itself.

**SQL Questions:**

1. Select all reviews that have a box\_ref\_id equal to **2** and a **rating** greater than **2**. (Expect 2 results.)

SELECT \* FROM reviews

WHERE box\_ref\_id = 2

AND rating > 2

1. Update the box with the **box\_id** of **3** to have an **about** of **Local produce delivered to your doorstep**.

UPDATE boxes

SET about = ‘Local produce delivered to your doorstep’

WHERE box\_id = 3

1. Add these two reviews for **Farm To People**. You may use multiple SQL statements. (Note: **box\_ref\_id** is a foreign key reference to **boxes**. This is a many-to-one relationship: many reviews to one box.)
   * title: **Thank you**, review: **Makes me happy and healthy**, rating: **5**
   * title: **Soggy**, review: **All the veggies were soggy**, rating: **2**

INSERT INTO reviews (title, review, rating, box\_ref\_id)

VALUES (‘Thank you’, ‘Makes me happy and healthy’, 5, 3)

INSERT INTO reviews (title, review, rating, box\_ref\_id)

VALUES (‘Soggy’, ‘All the veggies were soggy’, 2, 3)

ALTERNATIVELY (adding multiple)

INSERT INTO reviews (title, review, rating, box\_ref\_id)

VALUES (‘Thank you’, ‘Makes me happy and healthy’, 5, 3), (‘Soggy’, ‘All the veggies were soggy’, 2, 3)

1. In one SQL statement, get the brand, title, review, and rating of all boxes that have a rating **less than 4**. (Include ***only*** the **brand**, **title**, **review**, and **rating** columns in the result.) Here are the expected results (assuming you added the reviews from #3 above)...

|  |  |  |  |
| --- | --- | --- | --- |
| **brand** | **title** | **review** | **rating** |
| "Hello Fresh" | "Hello Fresh is trash" | "I found trash in the box" | 1 |
| "Butcher Box" | "Disaster" | "Parasites everywhere" | 1 |
| "Butcher Box" | "Mediocre at best" | "Costco has better meats" | 3 |
| "Farm To People" | "Soggy" | "All the veggies were soggy" | 2 |

SELECT brand, title, review, rating

FROM boxes

INNER JOIN reviews

ON boxes.box\_id = reviews.box\_ref\_id

WHERE reviews.rating < 4

**PART 2 - MONGODB**

**Submission:**

In the provided JavaScript file part2-answers.js, paste the MongoDB command for each of the six numbered bullets below. You do not need to submit the results of the queries, just the command itself.

**MongoDB Questions:**

1. Using one MongoDB command, add the following data to a collection named inventory. Let MongoDB generate IDs for you. (You won't lose points for typos in the data.)

|  |  |  |  |
| --- | --- | --- | --- |
| **name** | **department** | **quantity** | **price** |
| Toothpaste | personal care | 10 | 4 |
| Toothbrush | personal care | 56 | 7 |
| Shampoo | personal care | 41 | 9 |
| Soap | personal care | 44 | 5 |
| Eggs | grocery | 100 | 4 |
| Milk | grocery | 20 | 2 |
| Cheese | grocery | 30 | 5 |

db.inventory.insertMany([

{ “name”: “Toothpaste”, “department”: “personal care”, “quantity”: 10, “price”: 4 },

{ “name”: “Toothbrush”, “department”: “personal care”, “quantity”: 56, “price”: 7 },

{ “name”: “Shampoo”, “department”: “personal care”, “quantity”: 41, “price”: 9 },

{ “name”: “Soap”, “department”: “personal care”, “quantity”: 44, “price”: 5 },

{ “name”: “Eggs”, “department”: “grocery”, “quantity”: 100, “price”: 4 },

{ “name”: “Milk”, “department”: “grocery”, “quantity”: 20, “price”: 2 },

{ “name”: “Cheese”, “department”: “grocery”, “quantity”: 30, “price”: 5 }

]);

1. Subtract 5 from quantity for everything that has a name starting with "Tooth". Use a regular expression.

db.inventory.updateMany( { name: /^Tooth/ }, { $inc: { quantity: -5 } } );

1. Find all items where quantity is less than or equal to 20 or greater than 75. (The result should be Toothpaste, Eggs, and Milk.)

db.inventory.find( { $or: [ { quantity: { $lte: 20 } }, { quantity: { $gt: 20 } } ] } );

1. Find the top 3 most expensive items, sorted by price.

db.inventory.aggregate([ { $sort: { price: -1 } }, { $limit: 3 } ]);

1. Set the quantity for Shampoo to 50. Reference the document by its ID.

db.inventory.updateOne({ name: db.inventory.findOne( { name: ‘Shampoo’ }.\_id ), { $set: { quantity: 50 } } });

1. Delete all the items in the grocery department.

db.inventory.deleteMany( { department: ‘grocery’ } );

**CHECK YOUR SOLUTION**

1. SELECT \* FROM reviews WHERE box\_ref\_id = 2 AND rating > 2;
2. UPDATE boxes SET about = 'Local produce delivered to your doorstep'

WHERE box\_id = 3;

1. INSERT INTO reviews (title, review, rating, box\_ref\_id)

VALUES ('Thank you', 'Makes me happy and healthy', 5, 3);  
INSERT INTO reviews (title, review, rating, box\_ref\_id)

VALUES ('Soggy', 'All the veggies were soggy', 2, 3);

1. SELECT boxes.brand, reviews.title, reviews.review, reviews.rating FROM reviews

JOIN boxes ON reviews.box\_ref\_id = boxes.box\_id

WHERE reviews.rating < 4;

 - or -

SELECT brand, title, review, rating FROM reviews

JOIN boxes ON reviews.box\_ref\_id = boxes.box\_id

WHERE rating < 4;

1. db.inventory.insertMany([

  { name: "Toothpaste", department: "personal care", quantity: 10, price: 4 },

  { name: "Toothbrush", department: "personal care", quantity: 56, price: 7 },

  { name: "Shampoo", department: "personal care", quantity: 41, price: 9 },

  { name: "Soap", department: "personal care", quantity: 44, price: 5 },

  { name: "Eggs", department: "grocery", quantity: 100, price: 4 },

  { name: "Milk", department: "grocery", quantity: 20, price: 2 },

  { name: "Cheese", department: "grocery", quantity: 30, price: 5 },

]);

1. db.inventory.updateMany(

  { name: /^Tooth/ },

  { $inc: { quantity: -5 } }

);

1. db.inventory.find({ $or: [

  { quantity: { $lte: 20 } },

  { quantity: { $gt: 75 } }

]});

1. db.inventory.find().sort({ price: -1 }).limit(3);
2. db.inventory.update(

  { \_id: ObjectId("60b0f32ec8d2ff03b1ec070b") },

  { $set: { quantity: 50 } }

);

1. db.inventory.deleteMany(

  { department: "grocery" }

);