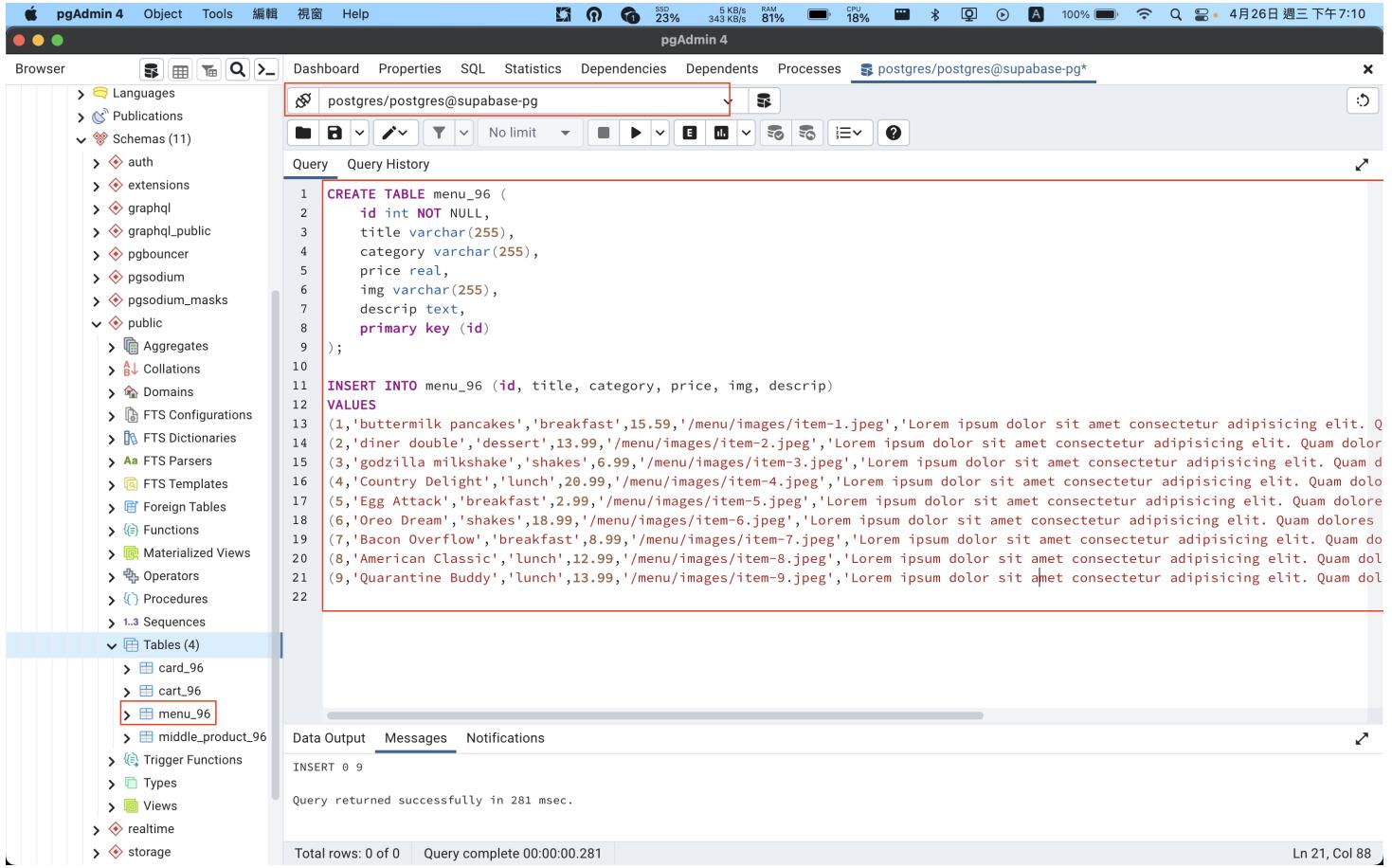


W011-P1: Midterm answer 6.a, 6.b



The screenshot shows the Supabase Table editor interface. On the left, there's a sidebar with navigation icons and a list of tables: card_96, cart_96, menu_96 (which is selected and highlighted with a red border), and middle_product_96. The main area displays a table titled "git-billy's Org / tku_210410196". The table has columns: id (int4), title (varchar), category (varchar), price (float4), img (varchar), and descrip (text). A warning message at the top says "WARNING: You are allowing anonymous access to your table. Enable Row Level Security". Below the table, there are buttons for Refresh, Filter, Sort, Insert, RLS is not enabled, API, Data, Definition, and Dismiss. At the bottom, there are page navigation controls (Page 1 of 1, 100 rows, 9 records).

W011-P2: Midterm answer 6.c, 7, 8

The screenshot shows the VS Code IDE interface. The Explorer sidebar on the left shows a project structure with folders like 1112-client-2n-c..., 1112-server..., bin, node_modules, public, routes, utils, data, 0426.sql, card_96.json, database.js, midprep_test..., midterm..., test-db.js, views, .env, .gitignore, app.js, package-lock..., and package.json. The file midterm_test_96.js is open in the editor. In the terminal tab, the command "node .bin/www" is run, connecting to SUPABASE PostgreSQL database database postgres on port 6543. The server starts on port 5001. The terminal output shows a JSON array of menu items:

```

menu_96: [{"id":1,"title":"buttermilk pancakes","category":"breakfast","price":15.59,"img":"/menu/images/item-1.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam."}, {"id":2,"title":"diner double","category":"dessert","price":13.99,"img":"/menu/images/item-2.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id":3,"title":"godzilla milkshake","category":"shakes","price":6.99,"img":"/menu/images/item-3.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id":4,"title":"Country Delight","category":"lunch","price":20.99,"img":"/menu/images/item-4.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id":5,"title":"Egg Attack","category":"breakfast","price":2.99,"img":"/menu/images/item-5.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id":6,"title":"Oreo Dream","category":"shakes","price":18.99,"img":"/menu/images/item-6.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id":7,"title":"Bacon Overflow","category":"breakfast","price":8.99,"img":"/menu/images/item-7.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id":8,"title":"American Classic","category":"lunch","price":12.99,"img":"/menu/images/item-8.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id":9,"title":"Quarantine Buddy","category":"lunch","price":13.99,"img":"/menu/images/item-9.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}]
category_96: [{"id":1,"title":"buttermilk pancakes","category":"breakfast","price":15.59,"img":"/menu/images/item-1.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam."}, {"id":5,"title":"Egg Attack","category":"breakfast","price":2.99,"img":"/menu/images/item-5.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id":7,"title":"Bacon Overflow","category":"breakfast","price":8.99,"img":"/menu/images/item-7.jpeg","descrip":"Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}]

```

The screenshot shows a developer environment with two main panes. The left pane is the Explorer view in VS Code, displaying the file structure of a project named 'TKU'. It includes files like 'apiMidtermRouter_96.js', 'w11-p2-3.png', 'routes.js', 'index.js', 'users.js', 'card_96.js', 'card2_96.js', 'midprep_xx.js', 'utils.js', 'views.js', '.env', '.gitignore', 'app.js', 'package-lock.json', and 'package.json'. The right pane shows the browser output at 'localhost:5001/api/mid_96/me...'. The browser displays a JSON response with 10 items, each representing a menu item with fields: id, title, category, price, img, and descrip. The first few items are:

```
[{"id": 1, "title": "buttermilk pancakes", "category": "breakfast", "price": 15.59, "img": "/menu/images/item-1.jpeg", "descrip": "Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam."}, {"id": 2, "title": "Oreo Dream", "category": "shakes", "price": 18.99, "img": "/menu/images/item-6.jpeg", "descrip": "Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id": 3, "title": "Bacon Overflow", "category": "breakfast", "price": 8.99, "img": "/menu/images/item-7.jpeg", "descrip": "Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id": 4, "title": "American Classic", "category": "lunch", "price": 12.99, "img": "/menu/images/item-8.jpeg", "descrip": "Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id": 5, "title": "Quarantine Buddy", "category": "lunch", "price": 13.99, "img": "/menu/images/item-9.jpeg", "descrip": "Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}]
```

The screenshot shows a developer's environment with two main windows. On the left is a code editor in VS Code displaying `apiMidtermRouter_96.js`. The code defines a Router for a database named `menu_96`, with routes for `/menu_96` and `/menu_96/:category`. The latter route uses a query to filter by category. On the right is a browser window showing the results of a GET request to `localhost:5001/api/mid_96/menu_96/breakfast`. The response is a JSON array of menu items, each with properties like title, category, price, and description. Three specific items are highlighted with red boxes around their `category` fields: "buttermilk pancakes", "Egg Attack", and "Bacon Overflow", all categorized as "breakfast".

```
var express = require('express');
var router = express.Router();

const db = require('../utils/database');

router.get('/menu_96', async function (req, res, next) {
  try {
    const results = await db.query('select * from menu_96');
    res.json(results.rows);
  } catch (error) {
    console.log(error);
  }
});

router.get('/menu_96/:category', async function (req, res, next) {
  try {
    let results = await db.query('select * from menu_96 where category = $1', [
      req.params.category,
    ]);
    console.log('cat_id data', JSON.stringify(results.rows));
    res.json(results.rows);
  } catch (error) {
    console.log(error);
  }
});

module.exports = router;
```

```
[{"id": 1, "title": "buttermilk pancakes", "category": "breakfast", "price": 15.59, "img": "/menu/images/item-1.jpeg", "descrip": "Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam."}, {"id": 5, "title": "Egg Attack", "category": "breakfast", "price": 2.99, "img": "/menu/images/item-5.jpeg", "descrip": "Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}, {"id": 7, "title": "Bacon Overflow", "category": "breakfast", "price": 8.99, "img": "/menu/images/item-7.jpeg", "descrip": "Lorem ipsum dolor sit amet consectetur adipisicing elit. Quam dolores ut iusto quas quia dignissimos ullam. Enim voluptas, expedita"}]
```

W011-P3: Show Midprep answers of node server with context for route /midp_node_context_xx

```
1112-client-2n-card-demo-96 > src > context > DemoContext_xx.js > [e] DemoProvider_xx
1  import React, { useContext, useReducer, useEffect } from "react";
2
3  import axios from "axios";
4
5  import DemoReducer_xx from "./DemoReducer_xx";
6
7  import { supabase } from "../db/clientSupabase";
8
9  // let api_midprep_url = 'http://localhost:5000/api/midprep_xx/overview2_xx';
10 // let api_midterm_url = 'http://localhost:5000/api/mid_xx/menu_xx';
11
12 let api_midprep_url = 'http://localhost:5001/api/midprep_96/overview2_96';
13
14 const initialState = {
15   phname: "ChungChun Wang",
16   pid: "210810196",
17   blogs: [],
18   blogs2: [],
19   data1: [],
20   data2: [],
21   menu: [],
22 };
23
24 const DemoContext_xx = React.createContext();
25
26 const DemoProvider_xx = ({ children }) => {
27   const [state, dispatch] = useReducer(DemoReducer_xx, initialState);
28
29   const fetchProductDataFromNodeServer = async () => {
30     try {
31       const results = await axios.get(api_midprep_url);
32       console.log('product data', results.data);
33       dispatch({ type: 'GET_PRODUCTS_NODE_SUCCESS', payload: results.data });
34     } catch (error) {
35       console.log(error);
36     }
37   };
38
39   useEffect(() => {
40     fetchProductDataFromNodeServer();
41   }, []);
42
43   const fetchMenuDataFromNodeServer = async (filter = '') => {
44     try {
45       const results = await axios.get(`${api_midterm_url}/${filter}`);
46       console.log('menu data', results.data);
47       dispatch({ type: 'GET_MENU_NODE_SUCCESS', payload: results.data });
48     } catch (error) {
49       console.log(error);
50     }
51   };
52
53   // useEffect(() => {
54   //   fetchMenuDataFromNodeServer();
55   // }, []);
56
57   // const changeMenuFilter = (filter) => {
58   //   console.log('filter', filter);
59   //   // fetchMenuDataFromNodeServer(filter);
60 };
61
62   return children(state, dispatch);
63 }
64
65
66 export default DemoProvider_xx;
```

```
1112-client-2n-card-demo-96 > src > context > DemoReducer_xx.js > [e] filter
1  const DemoReducer_xx = (state, action) => {
2    if (action.type === 'GET_BLOGS_SUPABASE_SUCCESS') {
3      return { ...state, blogs: action.payload };
4    }
5
6    if (action.type === 'GET_PRODUCTS_NODE_SUCCESS') {
7      return {
8        ...state,
9        data1: action.payload.data1,
10       data2: action.payload.data2,
11     };
12   }
13
14   if (action.type === 'GET_MENU_NODE_SUCCESS') {
15     return {
16       ...state,
17       menu: action.payload,
18     };
19   }
20
21   if (action.type === 'CHANGE_MENU_FILTER') {
22     const filter = action.payload === 'all' ? '' : action.payload;
23     return {
24       ...state,
25       menu_filter: filter,
26     };
27   }
28
29   export default DemoReducer_xx;
30
```

show how to get data1, data2 from api, and store with useContext and useReducer

The screenshot shows two code editor panes side-by-side. On the left is `DemoContext_xx.js` and on the right is `ProductsNodeServerContextPage_xx.js`. Both files are part of a React application structure.

`DemoContext_xx.js` contains:

```
1 import React, { useState, useReducer, useEffect } from "react";
2 import axios from "axios";
3
4 import DemoReducer_xx from "./DemoReducer_xx";
5
6 import { supabase } from "../db/clientSupabase";
7
8 // let api_midprep_url = "http://localhost:5008/api/midprep_xx/overview2_xx";
// let api_midterm_url = "http://localhost:5001/api/mid_xx/menu_xx";
9
10 let api_midprep_url = "http://localhost:5001/api/midprep_96/overview2_96";
11
12 const initialState = {
13   pName: "ChungChun Wang",
14   pid: "210410196",
15   blogs: [],
16   blogs2: [],
17   data1: [],
18   data2: [],
19   menu: []
20 };
21
22 const DemoContext_xx = React.createContext();
23
24 const DemoProvider_xx = ({ children }) => {
25   const [state, dispatch] = useReducer(DemoReducer_xx, initialState);
26
27   const fetchProductDataFromNodeServer = async () => {
28     try {
29       const results = await axios.get(api_midprep_url);
30       console.log("product data", results.data);
31       dispatch({ type: 'GET_PRODUCTS_NODE_SUCCESS', payload: results.data });
32     } catch (error) {
33       console.log(error);
34     }
35   };
36
37   useEffect(() => {
38     fetchProductDataFromNodeServer();
39   }, []);
40
41   return (
42     // <DemoContext_xx.Provider value={{ ... state, changeMenuFilter }}>
43     <DemoContext_xx.Provider value={{ ... state }}>
44       {children}
45     </DemoContext_xx.Provider>
46   );
47
48   const useDemoContext_xx = () => {
49     return useContext(DemoContext_xx);
50   };
51
52   export { DemoProvider_xx, useDemoContext_xx };
53 }
```

`ProductsNodeServerContextPage_xx.js` contains:

```
1 import { useState, useEffect } from "react";
2 // import axios from "axios";
3
4 import Wrapper from "../../../../assets/wrapper/midprep_xx/Products.scss.xx";
5
6 import { useDemoContext_xx } from "../../../../context/DemoContext_xx";
7
8 const ProductsNodeServerContextPage_xx = () => {
9   const [pName, pid, data1, data2, blogs] = useDemoContext_xx();
10
11   console.log()
12
13   return (
14     <Wrapper>
15       <div className="shop-page">
16         <div className="section-title">
17           <h2>Fetch Products From Node Server Using Context</h2>
18         </div>
19         <div className="collection-overview">
20           <div className="collection-preview">
21             <h3>Men's</h3>
22             <div className="menPreview">
23               <div>
24                 {data1.map(item) => (
25                   <div>
26                     <img alt={item}>
27                     <div>
28                       {item}
29                     </div>
30                   </div>
31                 )}
32               </div>
33             </div>
34           </div>
35           <div className="collection-preview">
36             <h3>Hats</h3>
37             <div className="hatsPreview">
38               <div>
39                 {data2.map(item) => (
40                   <div>
41                     <img alt={item}>
42                     <div>
43                       {item}
44                     </div>
45                   </div>
46                 )}
47               </div>
48             </div>
49           </div>
50         </div>
51       </div>
52     </Wrapper>
53   );
54
55   export default ProductsNodeServerContextPage_xx;
56 }
```

show how to get data1, data2 from context in ProductsNodeServerContextPage_xx

137f0e3 Billy Wed Apr 26 23:48:46 2023 +0800 w11