

# CPSC 304 Project Cover Page

Milestone #: 4

Date: August 5th, 2024

Group Number: 31

Name	Student Number	CS Alias (Userid)	Preferred Email Address
Leyang Pan	93460962	yang0526	yangplypan@gmail.com
Binjie Ye	29415965	yebinjie	yebinjie2021@163.com
Cheryl Chen	44983922	cchen70	chen.q.cheryl@gmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your email address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia.

## Repository Link:

[https://github.students.cs.ubc.ca/CPSC304-2024S-T2/project\\_i1q4s\\_o8n4r\\_t4k7v](https://github.students.cs.ubc.ca/CPSC304-2024S-T2/project_i1q4s_o8n4r_t4k7v)

Note: Some functionalities may not be available on Safari, please open the frontend with Google Chrome.

Our application is divided into two folders: frontend and backend

If one wants to test our application, they need to run them separately.

Backend:

- ssh into the remote student server
- clone the project into the remote machine
- direct to the backend folder using terminal
- "npm install" all the packages required
- create a .env file at the backend root directory and enter all the required information (the same as tutorial 2)
- "sh remote-start.sh" once all the packages are installed
- open a new local terminal and direct to the backend folder
- run ".\scripts\win\server-tunnel.cmd" if using windows
- enter the cwl ID and password
- the terminal should then prompt you the port number where the backend is running

Frontend:

- clone the project into a local machine
- open the terminal and direct to the frontend folder
- "npm install" all the packages required
- create a .env.local file at the frontend root directory and create one variable:
  - REACT\_APP\_API\_URL=http://localhost:[port]
- and change the port number to the one specified by the backend terminal
- Note: whenever restarting the backend, and if the port is changed, one needs to change the port number in this file and restart the frontend server
- "npm start" the server once all packages are installed
- by default, the server should be running at [localhost:3000](#)

## Brief Summary:

This project models [the Belcarra Beachkeepers summer program](#), in which volunteers conduct research on the crabs at a local provincial park, as well as educate park visitors about conservation. Entities include volunteers, supervisors, park visitors and crabbers, their crabs and traps, and logistics like roles and shifts. This project is intended to help staff schedule shifts and manage scientific data about the crabs and visitors, as well as make the research insights publicly available.

Users can login as either a Volunteer or a Supervisor, which sends them to their respective Dashboards. Their Dashboard loads basic statistics about them, including when their next shift is, the number of crabs caught/studied, and number of visitors they've interacted with.

From here, both volunteers and supervisors can switch to the Calendar page through the top menu bar. On the Calendar page, supervisors can create and delete shifts, while volunteers can sign up for certain roles on certain shifts.

Volunteers can uniquely navigate to the Upload page, where they can log crab, trap, and park visitor interaction data, which will update the database and be reflected on the General Insights and Crab Insights pages.

General Insights and Crab Insights are two pages that are also accessible to the public, so no login is necessary. Public users on the landing page can click “Guest? See our public page!” to be redirected to the public page. Here, they can perform most of the queries, such as join, projection, and all kinds of aggregations.

## Script:

Our script is named as: [beachkeepers.sql](#)

Located inside: [repository root/backend/beachkeepers.sql](#)

This script will drop & create all the tables inside the schema.

To run this script, we have two ways:

1. Login to sqlplus on the remote server and manually run the script by start `beachkeepers.sql` (recommended)
2. Initialize the remote server and go to: [localhost:\[port#\]/initialize-beachkeepers](#)  
This will automatically execute the file and populate the data, but sometimes it might fail due to dropping non-existent tables.

## Schema:

We have made slight modifications to our schema, such as renaming and deleting unimportant keys that do not impact the queries we implement, and altering the attribute type of SHIFT's highTide and lowTide from TIME to CHAR, as TIME is not supported by Oracle SQL. One major change we made is to add a password attribute to both Supervisor and Volunteer tables to support the login and sign up functionalities.

## Final Schema: Changes specified in red

Supervisor(employeeID: INTEGER, firstName: VARCHAR, lastName: VARCHAR, phone: VARCHAR, email: VARCHAR, **password: VARCHAR**)

- Primary key = employeeID, **combination of firstName and lastName needs to be unique**  
**Added password for the signup and login functionality**

Volunteer(volunteerID: INTEGER, firstName: VARCHAR, lastName: VARCHAR, experience: INTEGER, **employeeID**: INTEGER, **password: VARCHAR**)

- Primary key = volunteerID, employeeID is a FK to Supervisor and is NOT NULL, **combination of firstName and lastName needs to be unique**  
**Added password for the signup and login functionality**

Role(roleName: VARCHAR, date: date, difficulty: INTEGER, location: VARCHAR)

- Role is a weak entity that relies on Shift, with partial key = roleName, and primary key = (roleName, date), where date is a FK to Shift and is NOT NULL  
**Position(positionname: VARCHAR, shift\_date: DATE, difficulty: INT, pos\_location: VARCHAR)**  
**Only made changes to names, e.g, Role → Position, date → shift\_date**

Registers(**volunteerID**: INTEGER, **positionname**: VARCHAR, **shift\_date**: date)

- Registers is a separate table because it represents many-to-many, primary key = (volunteerID, roleName, date), volunteerID is a FK to Volunteer, roleName and date are FKs to Role (two attributes because Role is a weak entity with a partial key)  
**Only made changes to names, e.g, rolename → positionname**

Shift(date: date, lowTide: **CHAR**, highTide: **CHAR**)

- Primary key: date  
**There is not a TIME domain type in oracle sql, used CHAR instead.**

ParkVisitor(phone: VARCHAR, firstName: VARCHAR, **age**: VARCHAR)

- Primary key: phone  
**Only made changes to name, age only specifies age group, i.e, 'Youth' or 'Adult'**

## University of British Columbia, Vancouver

### Department of Computer Science

---

Crabber(phone: VARCHAR, licenseNum: INTEGER, **volunteerID**: INTEGER)

- Primary key = phone (also a FK as this is superclass's PK), volunteerID is a FK to Volunteer and is NOT NULL, licenseNum must be UNIQUE

Tourist(phone: VARCHAR, **pref\_language**: VARCHAR, homeCountry: VARCHAR)

- Primary key = phone (also a FK as this is superclass's PK)

Only made changes to name

Interacts(**volunteerID**: INTEGER, phone: VARCHAR, topic: VARCHAR)

- Interacts is a separate table because it represents many-to-many, primary key = (volunteerID, phone), volunteerID is a FK to Volunteer, phone is a FK to ParkVisitor

Trap(trapID: INTEGER, baitType: VARCHAR, trapType: VARCHAR, location: VARCHAR, successRate: DOUBLE, **phone**: VARCHAR)

- Primary key = trapID, phone is a FK to Crabber and is NOT NULL

Trap(trapID: INTEGER, baitType: VARCHAR, trapType: VARCHAR, trap\_location: VARCHAR, **phone**: VARCHAR)

successRate didn't contribute to any relevant queries and isn't an attribute for the actual Beachkeepers study, so removed to reduce development workload.

Crab(crab#: INTEGER, species: VARCHAR, sex: CHAR(1), shellCondition: VARCHAR, size: INTEGER, injury: CHAR(1), legalSize: INTEGER, takeHome: CHAR(1), **trapID**: INTEGER)

- Primary key = crab #, trapID is a FK to Trap and is NOT NULL

Crab(crab#: INTEGER, species: VARCHAR, sex: CHAR(1), **INTEGER**, injury: VARCHAR, **trapID**: INTEGER)

Since shellCondition, legalSize, and takeHome were redundant for most queries (other attributes did the same things), removed to reduce the development workload.

Studies(**volunteerID**: INTEGER, crab#: INTEGER)

- Studies is a separate table because it represents many-to-many, primary key = (volunteerID, crab #), volunteerID is a FK to Volunteer, crab # is a FK to Crab

## Schema Screenshots:

We have created two endpoints: [/tables](#) and [/tables/fetch?tableName=/tables](#) will fetch all the table names in our database, and [/tables/fetch?tableName=](#) will fetch all the tuples of the table with given table name

Note: Some schemas have been normalized into more tables.

Output of all tables through `"/tables/fetch?tableName="`:

Supervisor(employeeID: INTEGER, firstName: VARCHAR, lastName: VARCHAR, phone: VARCHAR, email: VARCHAR, password: VARCHAR)

```
{ "success": true, "rows":  
[{"EMPLOYEEID":1,"FIRSTNAME":"April","LASTNAME":"Rudgate","PASSWORD":"123","PHONE":"778-945-1049"},  
{"EMPLOYEEID":2,"FIRSTNAME":"Keely","LASTNAME":"Rangford","PASSWORD":"123","PHONE":"604-802-3581"},  
{"EMPLOYEEID":3,"FIRSTNAME":"Chris","LASTNAME":"Rahoney","PASSWORD":"123","PHONE":"778-381-4112"},  
{"EMPLOYEEID":4,"FIRSTNAME":"Danielle","LASTNAME":"Slark","PASSWORD":"123","PHONE":"604-202-0733"},  
{"EMPLOYEEID":5,"FIRSTNAME":"Rachel","LASTNAME":"Breen","PASSWORD":"123","PHONE":"778-515-8466"}]}
```

Email(firstName: VARCHAR, lastName: VARCHAR, email: VARCHAR)

```
{ "success": true, "rows":  
[{"FIRSTNAME":"April","LASTNAME":"Rudgate","EMAIL":"april.rudgate@gmail.com","EMPLOYEEID":1},  
{"FIRSTNAME":"Keely","LASTNAME":"Rangford","EMAIL":"keely.rangford@gmail.com","EMPLOYEEID":2},  
{"FIRSTNAME":"Chris","LASTNAME":"Rahoney","EMAIL":"chris.rahoney@gmail.com","EMPLOYEEID":3},  
{"FIRSTNAME":"Danielle","LASTNAME":"Slark","EMAIL":"danielle.slark@gmail.com","EMPLOYEEID":4},  
{"FIRSTNAME":"Rachel","LASTNAME":"Breen","EMAIL":"rachel.breen@gmail.com","EMPLOYEEID":5}]}
```

Volunteer(volunteerID: INTEGER, firstName: VARCHAR, lastName: VARCHAR, experience: INTEGER, employeeID: INTEGER, password: VARCHAR)

```
{ "success": true, "rows":  
[{"VOLUNTEERID":100,"FIRSTNAME":"Elayne","LASTNAME":"Lu","PASSWORD":"123","EXPERIENCE":9,"EMPLOYEEID":1},  
{"VOLUNTEERID":101,"FIRSTNAME":"Lisa","LASTNAME":"Kooz","PASSWORD":"123","EXPERIENCE":14,"EMPLOYEEID":5},  
{"VOLUNTEERID":102,"FIRSTNAME":"Cheryl","LASTNAME":"Cheng","PASSWORD":"123","EXPERIENCE":10,"EMPLOYEEID":5},  
{"VOLUNTEERID":103,"FIRSTNAME":"Calum","LASTNAME":"Lederat","PASSWORD":"123","EXPERIENCE":4,"EMPLOYEEID":2},  
{"VOLUNTEERID":104,"FIRSTNAME":"Eric","LASTNAME":"Dunce","PASSWORD":"123","EXPERIENCE":5,"EMPLOYEEID":3},  
{"VOLUNTEERID":105,"FIRSTNAME":"Jordin","LASTNAME":"Thumper","PASSWORD":"123","EXPERIENCE":5,"EMPLOYEEID":3},  
{"VOLUNTEERID":106,"FIRSTNAME":"Bailey","LASTNAME":"Johnston","PASSWORD":"123","EXPERIENCE":7,"EMPLOYEEID":3},  
{"VOLUNTEERID":107,"FIRSTNAME":"Danny","LASTNAME":"Wang","PASSWORD":"123","EXPERIENCE":6,"EMPLOYEEID":4},  
{"VOLUNTEERID":108,"FIRSTNAME":"Kevin","LASTNAME":"Xu","PASSWORD":"123","EXPERIENCE":8,"EMPLOYEEID":2},  
{"VOLUNTEERID":109,"FIRSTNAME":"Brute","LASTNAME":"Yang","PASSWORD":"123","EXPERIENCE":11,"EMPLOYEEID":1},  
{"VOLUNTEERID":110,"FIRSTNAME":"Jazzy","LASTNAME":"Singh","PASSWORD":"123","EXPERIENCE":4,"EMPLOYEEID":1},  
{"VOLUNTEERID":111,"FIRSTNAME":"Ani","LASTNAME":"Rube","PASSWORD":"123","EXPERIENCE":2,"EMPLOYEEID":2},  
{"VOLUNTEERID":112,"FIRSTNAME":"Dora","LASTNAME":"Rose","PASSWORD":"123","EXPERIENCE":1,"EMPLOYEEID":5},  
{"VOLUNTEERID":113,"FIRSTNAME":"Kirsty","LASTNAME":"Fluver","PASSWORD":"123","EXPERIENCE":0,"EMPLOYEEID":5},  
{"VOLUNTEERID":114,"FIRSTNAME":"Orla","LASTNAME":"Sharma","PASSWORD":"123","EXPERIENCE":12,"EMPLOYEEID":4},  
{"VOLUNTEERID":115,"FIRSTNAME":"Uma","LASTNAME":"Smith","PASSWORD":"123","EXPERIENCE":3,"EMPLOYEEID":4}]}
```

Position(positionname: VARCHAR, shift\_date: DATE, difficulty: INT, pos\_location: VARCHAR)

```
"rows": [
  {
    "POSITIONNAME": "Measurer",
    "SHIFT_DATE": "2023-06-24T07:00:00.000Z",
    "DIFFICULTY": 3,
    "POS_LOCATION": "wharf"
  },
  {
    "POSITIONNAME": "Recorder",
    "SHIFT_DATE": "2023-07-09T07:00:00.000Z",
    "DIFFICULTY": 5,
    "POS_LOCATION": "wharf"
  },
  {
    "POSITIONNAME": "Handler",
    "SHIFT_DATE": "2024-06-30T07:00:00.000Z",
    "DIFFICULTY": 4,
    "POS_LOCATION": "wharf"
  },
  {
    "POSITIONNAME": "Info tent",
    "SHIFT_DATE": "2024-07-01T07:00:00.000Z",
    "DIFFICULTY": 1,
    "POS_LOCATION": "picnic area"
  },
  {
    "POSITIONNAME": "Measurer",
    "SHIFT_DATE": "2024-07-13T07:00:00.000Z",
    "DIFFICULTY": 3,
    "POS_LOCATION": "wharf"
  }
]
```

Registers(volunteerID: INTEGER, positionname: VARCHAR, shift\_date: date)

```
{
  "success": true,
  "rows": [
    {
      "VOLUNTEERID": 100,
      "POSITIONNAME": "Handler",
      "SHIFT_DATE": "2024-06-30T07:00:00.000Z"
    },
    {
      "VOLUNTEERID": 101,
      "POSITIONNAME": "Measurer",
      "SHIFT_DATE": "2024-07-13T07:00:00.000Z"
    },
    {
      "VOLUNTEERID": 102,
      "POSITIONNAME": "Info tent",
      "SHIFT_DATE": "2024-07-01T07:00:00.000Z"
    },
    {
      "VOLUNTEERID": 103,
      "POSITIONNAME": "Measurer",
      "SHIFT_DATE": "2023-06-24T07:00:00.000Z"
    },
    {
      "VOLUNTEERID": 104,
      "POSITIONNAME": "Measurer",
      "SHIFT_DATE": "2023-06-24T07:00:00.000Z"
    }
  ]
}
```

Shift(date: date, lowTide: CHAR, highTide: CHAR)

```
"rows": [
  {
    "SHIFT_DATE": "2023-06-24T07:00:00.000Z",
    "LOWTIDE": "10:34:23",
    "HIGHTIDE": "21:59:00"
  },
  {
    "SHIFT_DATE": "2023-07-09T07:00:00.000Z",
    "LOWTIDE": "10:39:48",
    "HIGHTIDE": "22:11:08"
  },
  {
    "SHIFT_DATE": "2024-06-30T07:00:00.000Z",
    "LOWTIDE": "10:52:28",
    "HIGHTIDE": "23:18:47"
  },
  {
    "SHIFT_DATE": "2024-07-01T07:00:00.000Z",
    "LOWTIDE": "11:44:53",
    "HIGHTIDE": "24:42:12"
  },
  {
    "SHIFT_DATE": "2024-07-13T07:00:00.000Z",
    "LOWTIDE": "12:26:27",
    "HIGHTIDE": "00:33:06"
  }
]
```

ParkVisitor(phone: VARCHAR, firstName: VARCHAR, age: VARCHAR)

```
"rows": [
  {
    "PHONE": "778-284-9134",
    "FIRSTNAME": "Maxim",
    "AGE": "Youth"
  },
  {
    "PHONE": "604-038-1435",
    "FIRSTNAME": "Layla",
    "AGE": "Adult"
  },
  {
    "PHONE": "604-678-4319",
    "FIRSTNAME": "Raymond",
    "AGE": "Adult"
  },
  {
    "PHONE": "224-649-9683",
    "FIRSTNAME": "Alan",
    "AGE": "Youth"
  },
  {
    "PHONE": "316-274-3485",
    "FIRSTNAME": "Edouard",
    "AGE": "Adult"
  },
  {
    "PHONE": "856-462-3021",
    "FIRSTNAME": "Ivan",
    "AGE": "Youth"
  },
  {
    "PHONE": "982-017-0112",
    "FIRSTNAME": "Celeste",
    "AGE": "Adult"
  },
  {
    "PHONE": "245-924-5203",
    "FIRSTNAME": "Elisa",
    "AGE": "Adult"
  }
],
```

```
{
  "PHONE": "374-939-2754",
  "FIRSTNAME": "Chloe",
  "AGE": "Adult"
},
{
  "PHONE": "984-921-5234",
  "FIRSTNAME": "Dave",
  "AGE": "Adult"
},
{
  "PHONE": "185-274-5245",
  "FIRSTNAME": "Trevor",
  "AGE": "Adult"
},
{
  "PHONE": "256-397-9103",
  "FIRSTNAME": "Favian",
  "AGE": "Adult"
},
{
  "PHONE": "642-174-8903",
  "FIRSTNAME": "Rob",
  "AGE": "Adult"
},
{
  "PHONE": "876-924-0982",
  "FIRSTNAME": "Russell",
  "AGE": "Adult"
},
{
  "PHONE": "275-431-0470",
  "FIRSTNAME": "Barbara",
  "AGE": "Youth"
},
{
  "PHONE": "389-134-7421",
  "FIRSTNAME": "Carol",
  "AGE": "Youth"
},
{
  "PHONE": "130-984-3497",
  "FIRSTNAME": "Lee",
  "AGE": "Youth"
}
```

Crabber(phone: VARCHAR, licenseNum: INTEGER, volunteerID: INTEGER)

```
{
  "PHONE": "778-284-9134",
  "LICENSENUM": 825492638,
  "VOLUNTEERID": 101
},
{
  "PHONE": "604-038-1435",
  "LICENSENUM": 740274927,
  "VOLUNTEERID": 102
},
{
  "PHONE": "604-678-4319",
  "LICENSENUM": 312436442,
  "VOLUNTEERID": 104
},
{
  "PHONE": "316-274-3485",
  "LICENSENUM": 628365816,
  "VOLUNTEERID": 102
},
{
  "PHONE": "245-924-5203",
  "LICENSENUM": 528947552,
  "VOLUNTEERID": 102
},
,
```

```
{
  "PHONE": "374-939-2754",
  "LICENSENUM": 467898763,
  "VOLUNTEERID": 110
},
{
  "PHONE": "984-921-5234",
  "LICENSENUM": 398475253,
  "VOLUNTEERID": 115
},
{
  "PHONE": "185-274-5245",
  "LICENSENUM": 824779340,
  "VOLUNTEERID": 109
},
{
  "PHONE": "256-397-9103",
  "LICENSENUM": 143251431,
  "VOLUNTEERID": 108
},
{
  "PHONE": "642-174-8903",
  "LICENSENUM": 165413437,
  "VOLUNTEERID": 107
},
,
```



Tourist(phone: VARCHAR, pref\_language: VARCHAR, homeCountry: VARCHAR)

```
{
  "PHONE": "224-649-9683",
  "PREF_LANGUAGE": "English",
  "HOMECOUNTRY": "Canada"
},
{
  "PHONE": "316-274-3485",
  "PREF_LANGUAGE": "French",
  "HOMECOUNTRY": "Canada"
},
{
  "PHONE": "856-462-3021",
  "PREF_LANGUAGE": "Spanish",
  "HOMECOUNTRY": "Chile"
},
{
  "PHONE": "982-017-0112",
  "PREF_LANGUAGE": "Mandarin",
  "HOMECOUNTRY": "China"
},
{
  "PHONE": "876-924-0982",
  "PREF_LANGUAGE": "English",
  "HOMECOUNTRY": "Canada"
},
{
  "PHONE": "275-431-0470",
  "PREF_LANGUAGE": "Korean",
  "HOMECOUNTRY": "South Korea"
},
{
  "PHONE": "389-134-7421",
  "PREF_LANGUAGE": "Korean",
  "HOMECOUNTRY": "South Korea"
},
{
  "PHONE": "130-984-3497",
  "PREF_LANGUAGE": "Hindi",
  "HOMECOUNTRY": "India"
}
```

Interacts(volunteerID: INTEGER, phone: VARCHAR, topic: VARCHAR)

```
{"success":true,"rows":[{"VOLUNTEERID":104,"PHONE":"224-649-9683","TOPIC":"Starfish"},
{"VOLUNTEERID":101,"PHONE":"316-274-3485","TOPIC":"Jellyfish bloom"},
{"VOLUNTEERID":100,"PHONE":"604-678-4319","TOPIC":"Sexing crabs"},
{"VOLUNTEERID":100,"PHONE":"604-038-1435","TOPIC":"Invasive crabs"},
{"VOLUNTEERID":105,"PHONE":"316-274-3485","TOPIC":"Ochre stars"},
{"VOLUNTEERID":107,"PHONE":"374-939-2754","TOPIC":"Leather stars"},
{"VOLUNTEERID":112,"PHONE":"982-017-0112","TOPIC":"Eating algae"},
{"VOLUNTEERID":110,"PHONE":"185-274-5245","TOPIC":"Proper bait disposal"},
{"VOLUNTEERID":109,"PHONE":"984-921-5234","TOPIC":"Getting a sports fishing license"},
{"VOLUNTEERID":102,"PHONE":"876-924-0982","TOPIC":"Crabbing success these days"},
{"VOLUNTEERID":103,"PHONE":"256-397-9103","TOPIC":"Jellyfish bloom"},
{"VOLUNTEERID":106,"PHONE":"984-921-5234","TOPIC":"Dead seal pup"},
{"VOLUNTEERID":108,"PHONE":"256-397-9103","TOPIC":"Orcas swimming up Indian Arm"},
{"VOLUNTEERID":111,"PHONE":"316-274-3485","TOPIC":"Boat docking area"},
{"VOLUNTEERID":113,"PHONE":"856-462-3021","TOPIC":"Boat docking area"},
{"VOLUNTEERID":114,"PHONE":"389-134-7421","TOPIC":"Jellyfish bloom"},
{"VOLUNTEERID":115,"PHONE":"642-174-8903","TOPIC":"Sexing crabs"},
{"VOLUNTEERID":107,"PHONE":"316-274-3485","TOPIC":"Local crab species"},
{"VOLUNTEERID":102,"PHONE":"984-921-5234","TOPIC":"Where is Cod Rock trail"},
{"VOLUNTEERID":102,"PHONE":"245-924-5203","TOPIC":"Jughead Island hike difficulty"},
{"VOLUNTEERID":105,"PHONE":"185-274-5245","TOPIC":"Jellyfish bloom"},
{"VOLUNTEERID":110,"PHONE":"275-431-0470","TOPIC":"Sexing crabs"},
{"VOLUNTEERID":101,"PHONE":"130-984-3497","TOPIC":"Getting a sports fishing license"},
{"VOLUNTEERID":104,"PHONE":"374-939-2754","TOPIC":"Jellyfish bloom"},
{"VOLUNTEERID":106,"PHONE":"778-284-9134","TOPIC":"Jellyfish bloom"},
{"VOLUNTEERID":114,"PHONE":"642-174-8903","TOPIC":"Illegal crabbing"}]}
```

## University of British Columbia, Vancouver

### Department of Computer Science

Trap(trapID: INTEGER, baitType: VARCHAR, trapType: VARCHAR, trap\_location: VARCHAR, phone: VARCHAR)

```
[{"TRAPID":1,"BAITTYPE":"Chicken","TRAPTYPE":"Circle","TRAP_LOCATION":"East","PHONE":"778-284-9134"}, {"TRAPID":2,"BAITTYPE":"Turkey","TRAPTYPE":"Clamshell","TRAP_LOCATION":"Main","PHONE":"604-038-1435"}, {"TRAPID":3,"BAITTYPE":"Turkey","TRAPTYPE":"Clamshell","TRAP_LOCATION":"North","PHONE":"604-678-4319"}, {"TRAPID":4,"BAITTYPE":"Pork","TRAPTYPE":"Box","TRAP_LOCATION":"South","PHONE":"642-174-8903"}, {"TRAPID":5,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"South","PHONE":"642-174-8903"}, {"TRAPID":6,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"East","PHONE":"256-397-9103"}, {"TRAPID":10,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"Main","PHONE":"256-397-9103"}, {"TRAPID":12,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"North","PHONE":"185-274-5245"}, {"TRAPID":14,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"East","PHONE":"374-939-2754"}, {"TRAPID":13,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"North","PHONE":"374-939-2754"}, {"TRAPID":15,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"South","PHONE":"185-274-5245"}, {"TRAPID":17,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"East","PHONE":"245-924-5203"}, {"TRAPID":8,"BAITTYPE":"Chicken","TRAPTYPE":"Circle","TRAP_LOCATION":"South","PHONE":"984-921-5234"}, {"TRAPID":26,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"East","PHONE":"984-921-5234"}, {"TRAPID":34,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"Main","PHONE":"245-924-5203"}, {"TRAPID":50,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"Main","PHONE":"316-274-3485"}, {"TRAPID":11,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"Main","PHONE":"316-274-3485"}, {"TRAPID":16,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"Main","PHONE":"316-274-3485"}, {"TRAPID":9,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"Main","PHONE":"316-274-3485"}]
```

Crab(crab#: INTEGER, species: VARCHAR, sex: CHAR(1), INTEGER, injury: VARCHAR, trapID: INTEGER)

```
{"success":true,"rows":[{"CRABID":1,"SPECIES":"Red rock","SEX":"M","CRAB_SIZE":108,"INJURY":"Y","TRAPID":5}, {"CRABID":2,"SPECIES":"Graceful","SEX":"F","CRAB_SIZE":59,"INJURY":"Y","TRAPID":2}, {"CRABID":3,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":103,"INJURY":"N","TRAPID":3}, {"CRABID":4,"SPECIES":"Graceful","SEX":"M","CRAB_SIZE":94,"INJURY":"N","TRAPID":3}, {"CRABID":5,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":87,"INJURY":"Y","TRAPID":4}, {"CRABID":6,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":81,"INJURY":"N","TRAPID":1}, {"CRABID":7,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":100,"INJURY":"N","TRAPID":1}, {"CRABID":8,"SPECIES":"Dungeness","SEX":"M","CRAB_SIZE":78,"INJURY":"N","TRAPID":1}, {"CRABID":9,"SPECIES":"Graceful","SEX":"F","CRAB_SIZE":70,"INJURY":"N","TRAPID":1}, {"CRABID":10,"SPECIES":"Dungeness","SEX":"M","CRAB_SIZE":83,"INJURY":"N","TRAPID":1}, {"CRABID":11,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":77,"INJURY":"N","TRAPID":1}, {"CRABID":12,"SPECIES":"Graceful","SEX":"F","CRAB_SIZE":77,"INJURY":"N","TRAPID":1}, {"CRABID":13,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":71,"INJURY":"N","TRAPID":1}, {"CRABID":14,"SPECIES":"Graceful","SEX":"M","CRAB_SIZE":77,"INJURY":"N","TRAPID":1}, {"CRABID":15,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":70,"INJURY":"N","TRAPID":1}, {"CRABID":16,"SPECIES":"Graceful","SEX":"F","CRAB_SIZE":57,"INJURY":"N","TRAPID":1}, {"CRABID":17,"SPECIES":"Graceful","SEX":"F","CRAB_SIZE":73,"INJURY":"Y","TRAPID":1}, {"CRABID":18,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":67,"INJURY":"N","TRAPID":1}, {"CRABID":19,"SPECIES":"Graceful","SEX":"F","CRAB_SIZE":76,"INJURY":"N","TRAPID":1}, {"CRABID":20,"SPECIES":"Graceful","SEX":"F","CRAB_SIZE":68,"INJURY":"N","TRAPID":1}, {"CRABID":21,"SPECIES":"Graceful","SEX":"F","CRAB_SIZE":73,"INJURY":"N","TRAPID":1}, {"CRABID":22,"SPECIES":"Dungeness","SEX":"M","CRAB_SIZE":74,"INJURY":"N","TRAPID":4}, {"CRABID":23,"SPECIES":"Dungeness","SEX":"M","CRAB_SIZE":68,"INJURY":"N","TRAPID":3}, {"CRABID":24,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":73,"INJURY":"N","TRAPID":3}, {"CRABID":25,"SPECIES":"Dungeness","SEX":"M","CRAB_SIZE":73,"INJURY":"N","TRAPID":2}, {"CRABID":26,"SPECIES":"Dungeness","SEX":"M","CRAB_SIZE":85,"INJURY":"N","TRAPID":2}, {"CRABID":27,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":67,"INJURY":"N","TRAPID":2}, {"CRABID":28,"SPECIES":"Graceful","SEX":"F","CRAB_SIZE":78,"INJURY":"N","TRAPID":6}, {"CRABID":29,"SPECIES":"Red rock","SEX":"F","CRAB_SIZE":85,"INJURY":"N","TRAPID":6}, {"CRABID":30,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":97,"INJURY":"N","TRAPID":6}, {"CRABID":31,"SPECIES":"Red rock","SEX":"M","CRAB_SIZE":81,"INJURY":"N","TRAPID":5}, {"CRABID":32,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":80,"INJURY":"N","TRAPID":5}, {"CRABID":33,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":74,"INJURY":"N","TRAPID":5}, {"CRABID":34,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":81,"INJURY":"N","TRAPID":5}, {"CRABID":35,"SPECIES":"Dungeness","SEX":"M","CRAB_SIZE":76,"INJURY":"Y","TRAPID":1}, {"CRABID":36,"SPECIES":"Red rock","SEX":"F","CRAB_SIZE":103,"INJURY":"N","TRAPID":1}, {"CRABID":37,"SPECIES":"Dungeness","SEX":"M","CRAB_SIZE":78,"INJURY":"N","TRAPID":1}, {"CRABID":38,"SPECIES":"Dungeness","SEX":"M","CRAB_SIZE":81,"INJURY":"N","TRAPID":1}, {"CRABID":39,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":87,"INJURY":"N","TRAPID":1}, {"CRABID":40,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":80,"INJURY":"N","TRAPID":1}, {"CRABID":41,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":80,"INJURY":"N","TRAPID":1}, {"CRABID":42,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":85,"INJURY":"N","TRAPID":1}, {"CRABID":43,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":91,"INJURY":"N","TRAPID":10}, {"CRABID":44,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":91,"INJURY":"N","TRAPID":10}, {"CRABID":45,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":95,"INJURY":"Y","TRAPID":10}, {"CRABID":46,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":91,"INJURY":"N","TRAPID":10}, {"CRABID":47,"SPECIES":"Graceful","SEX":"F","CRAB_SIZE":99,"INJURY":"N","TRAPID":10}, {"CRABID":48,"SPECIES":"Dungeness","SEX":"F","CRAB_SIZE":83,"INJURY":"N","TRAPID":10}, {"CRABID":49,"SPECIES":"Red rock","SEX":"F","CRAB_SIZE":116,"INJURY":"N","TRAPID":9}, {"CRABID":50,"SPECIES":"Graceful","SEX":"M","CRAB_SIZE":85,"INJURY":"N","TRAPID":9}]}
```



University of British Columbia, Vancouver  
Department of Computer Science

```
{ "CRABID": 51, "SPECIES": "Graceful", "SEX": "F", "CRAB_SIZE": 67, "INJURY": "Y", "TRAPID": 12 },
{ "CRABID": 52, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 84, "INJURY": "N", "TRAPID": 12 },
{ "CRABID": 53, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 92, "INJURY": "N", "TRAPID": 12 },
{ "CRABID": 54, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 96, "INJURY": "N", "TRAPID": 12 },
{ "CRABID": 55, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 99, "INJURY": "N", "TRAPID": 12 },
{ "CRABID": 56, "SPECIES": "Graceful", "SEX": "M", "CRAB_SIZE": 63, "INJURY": "N", "TRAPID": 12 },
{ "CRABID": 57, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 67, "INJURY": "N", "TRAPID": 12 },
{ "CRABID": 58, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 71, "INJURY": "N", "TRAPID": 10 },
{ "CRABID": 59, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 152, "INJURY": "N", "TRAPID": 10 },
{ "CRABID": 60, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 89, "INJURY": "N", "TRAPID": 10 },
{ "CRABID": 61, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 85, "INJURY": "N", "TRAPID": 10 },
{ "CRABID": 62, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 87, "INJURY": "N", "TRAPID": 10 },
{ "CRABID": 63, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 97, "INJURY": "Y", "TRAPID": 10 },
{ "CRABID": 64, "SPECIES": "Graceful", "SEX": "F", "CRAB_SIZE": 86, "INJURY": "Y", "TRAPID": 5 },
{ "CRABID": 65, "SPECIES": "Graceful", "SEX": "F", "CRAB_SIZE": 76, "INJURY": "Y", "TRAPID": 9 },
{ "CRABID": 66, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 77, "INJURY": "N", "TRAPID": 9 },
{ "CRABID": 67, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 76, "INJURY": "N", "TRAPID": 5 },
{ "CRABID": 68, "SPECIES": "Graceful", "SEX": "M", "CRAB_SIZE": 74, "INJURY": "N", "TRAPID": 5 },
{ "CRABID": 69, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 88, "INJURY": "N", "TRAPID": 10 },
{ "CRABID": 70, "SPECIES": "Graceful", "SEX": "M", "CRAB_SIZE": 83, "INJURY": "N", "TRAPID": 10 },
{ "CRABID": 71, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 94, "INJURY": "N", "TRAPID": 10 },
{ "CRABID": 72, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 95, "INJURY": "N", "TRAPID": 10 },
{ "CRABID": 73, "SPECIES": "Graceful", "SEX": "M", "CRAB_SIZE": 72, "INJURY": "N", "TRAPID": 9 },
{ "CRABID": 74, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 92, "INJURY": "N", "TRAPID": 14 },
{ "CRABID": 75, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 83, "INJURY": "Y", "TRAPID": 14 },
{ "CRABID": 76, "SPECIES": "Graceful", "SEX": "M", "CRAB_SIZE": 95, "INJURY": "N", "TRAPID": 16 },
{ "CRABID": 77, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 78, "INJURY": "N", "TRAPID": 16 },
{ "CRABID": 94, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 67, "INJURY": "N", "TRAPID": 13 },
{ "CRABID": 95, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 72, "INJURY": "N", "TRAPID": 13 },
{ "CRABID": 96, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 81, "INJURY": "N", "TRAPID": 13 },
{ "CRABID": 97, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 94, "INJURY": "N", "TRAPID": 15 },
{ "CRABID": 98, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 80, "INJURY": "N", "TRAPID": 15 },
{ "CRABID": 99, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 93, "INJURY": "N", "TRAPID": 15 },
{ "CRABID": 100, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 105, "INJURY": "N", "TRAPID": 15 },
{ "CRABID": 101, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 96, "INJURY": "N", "TRAPID": 15 },
{ "CRABID": 102, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 108, "INJURY": "N", "TRAPID": 15 },
{ "CRABID": 108, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 108, "INJURY": "N", "TRAPID": 17 },
{ "CRABID": 109, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 116, "INJURY": "N", "TRAPID": 17 },
{ "CRABID": 110, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 92, "INJURY": "N", "TRAPID": 17 },
{ "CRABID": 124, "SPECIES": "Graceful", "SEX": "F", "CRAB_SIZE": 75, "INJURY": "Y", "TRAPID": 8 },
{ "CRABID": 125, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 67, "INJURY": "N", "TRAPID": 8 },
{ "CRABID": 126, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 75, "INJURY": "Y", "TRAPID": 8 },
{ "CRABID": 173, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 94, "INJURY": "N", "TRAPID": 26 },
{ "CRABID": 174, "SPECIES": "Graceful", "SEX": "F", "CRAB_SIZE": 77, "INJURY": "Y", "TRAPID": 26 },
{ "CRABID": 180, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 90, "INJURY": "Y", "TRAPID": 34 },
{ "CRABID": 181, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 95, "INJURY": "N", "TRAPID": 50 },
{ "CRABID": 182, "SPECIES": "Graceful", "SEX": "M", "CRAB_SIZE": 80, "INJURY": "Y", "TRAPID": 50 },
{ "CRABID": 191, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 84, "INJURY": "N", "TRAPID": 11 },
{ "CRABID": 267, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 93, "INJURY": "N", "TRAPID": 8 },
{ "CRABID": 268, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 89, "INJURY": "N", "TRAPID": 13 } ] }
```

Studies(volunteerID: INTEGER, crab#: INTEGER)

```
{ "success": true, "rows": [ { "VOLUNTEERID": 100, "CRABID": 9 }, { "VOLUNTEERID": 100, "CRABID": 19 },
{ "VOLUNTEERID": 100, "CRABID": 34 }, { "VOLUNTEERID": 100, "CRABID": 42 }, { "VOLUNTEERID": 100, "CRABID": 54 },
{ "VOLUNTEERID": 100, "CRABID": 72 }, { "VOLUNTEERID": 100, "CRABID": 94 }, { "VOLUNTEERID": 100, "CRABID": 97 },
{ "VOLUNTEERID": 100, "CRABID": 101 }, { "VOLUNTEERID": 100, "CRABID": 109 }, { "VOLUNTEERID": 100, "CRABID": 174 },
{ "VOLUNTEERID": 101, "CRABID": 5 }, { "VOLUNTEERID": 101, "CRABID": 15 }, { "VOLUNTEERID": 101, "CRABID": 29 },
{ "VOLUNTEERID": 101, "CRABID": 39 }, { "VOLUNTEERID": 101, "CRABID": 45 }, { "VOLUNTEERID": 101, "CRABID": 51 },
{ "VOLUNTEERID": 101, "CRABID": 57 }, { "VOLUNTEERID": 101, "CRABID": 59 }, { "VOLUNTEERID": 101, "CRABID": 60 },
{ "VOLUNTEERID": 101, "CRABID": 66 }, { "VOLUNTEERID": 101, "CRABID": 110 }, { "VOLUNTEERID": 102, "CRABID": 16 },
{ "VOLUNTEERID": 102, "CRABID": 22 }, { "VOLUNTEERID": 102, "CRABID": 25 }, { "VOLUNTEERID": 102, "CRABID": 27 },
{ "VOLUNTEERID": 102, "CRABID": 30 }, { "VOLUNTEERID": 102, "CRABID": 37 }, { "VOLUNTEERID": 102, "CRABID": 47 },
{ "VOLUNTEERID": 102, "CRABID": 48 }, { "VOLUNTEERID": 102, "CRABID": 50 }, { "VOLUNTEERID": 102, "CRABID": 55 },
{ "VOLUNTEERID": 102, "CRABID": 61 }, { "VOLUNTEERID": 102, "CRABID": 74 }, { "VOLUNTEERID": 102, "CRABID": 124 },
{ "VOLUNTEERID": 102, "CRABID": 182 }, { "VOLUNTEERID": 103, "CRABID": 1 }, { "VOLUNTEERID": 103, "CRABID": 49 },
{ "VOLUNTEERID": 103, "CRABID": 55 }, { "VOLUNTEERID": 103, "CRABID": 60 }, { "VOLUNTEERID": 103, "CRABID": 62 },
{ "VOLUNTEERID": 103, "CRABID": 67 }, { "VOLUNTEERID": 103, "CRABID": 69 }, { "VOLUNTEERID": 103, "CRABID": 97 },
{ "VOLUNTEERID": 103, "CRABID": 98 }, { "VOLUNTEERID": 104, "CRABID": 2 }, { "VOLUNTEERID": 104, "CRABID": 4 },
{ "VOLUNTEERID": 104, "CRABID": 59 }, { "VOLUNTEERID": 104, "CRABID": 66 }, { "VOLUNTEERID": 104, "CRABID": 77 },
{ "VOLUNTEERID": 105, "CRABID": 24 }, { "VOLUNTEERID": 105, "CRABID": 31 }, { "VOLUNTEERID": 105, "CRABID": 64 },
```

```
{ "VOLUNTEERID":105,"CRABID":71},{ "VOLUNTEERID":105,"CRABID":95},{ "VOLUNTEERID":105,"CRABID":100},
{ "VOLUNTEERID":105,"CRABID":126},{ "VOLUNTEERID":105,"CRABID":181},{ "VOLUNTEERID":105,"CRABID":191},
{ "VOLUNTEERID":106,"CRABID":7},{ "VOLUNTEERID":106,"CRABID":13},{ "VOLUNTEERID":106,"CRABID":17},
{ "VOLUNTEERID":106,"CRABID":21},{ "VOLUNTEERID":106,"CRABID":24},{ "VOLUNTEERID":106,"CRABID":26},
{ "VOLUNTEERID":106,"CRABID":37},{ "VOLUNTEERID":106,"CRABID":48},{ "VOLUNTEERID":106,"CRABID":62},
{ "VOLUNTEERID":107,"CRABID":6},{ "VOLUNTEERID":107,"CRABID":11},{ "VOLUNTEERID":107,"CRABID":27},
{ "VOLUNTEERID":107,"CRABID":36},{ "VOLUNTEERID":107,"CRABID":40},{ "VOLUNTEERID":107,"CRABID":52},
{ "VOLUNTEERID":107,"CRABID":59},{ "VOLUNTEERID":107,"CRABID":72},{ "VOLUNTEERID":107,"CRABID":181},
{ "VOLUNTEERID":107,"CRABID":182},{ "VOLUNTEERID":108,"CRABID":6},{ "VOLUNTEERID":108,"CRABID":58},
{ "VOLUNTEERID":108,"CRABID":70},{ "VOLUNTEERID":108,"CRABID":77},{ "VOLUNTEERID":108,"CRABID":96},
{ "VOLUNTEERID":108,"CRABID":108},{ "VOLUNTEERID":109,"CRABID":67},{ "VOLUNTEERID":109,"CRABID":73},
{ "VOLUNTEERID":109,"CRABID":101},{ "VOLUNTEERID":109,"CRABID":173},{ "VOLUNTEERID":109,"CRABID":180},
{ "VOLUNTEERID":109,"CRABID":267},{ "VOLUNTEERID":110,"CRABID":8},{ "VOLUNTEERID":110,"CRABID":12},
{ "VOLUNTEERID":110,"CRABID":14},{ "VOLUNTEERID":110,"CRABID":33},{ "VOLUNTEERID":110,"CRABID":44},
{ "VOLUNTEERID":110,"CRABID":56},{ "VOLUNTEERID":110,"CRABID":95},{ "VOLUNTEERID":110,"CRABID":108},
{ "VOLUNTEERID":111,"CRABID":20},{ "VOLUNTEERID":111,"CRABID":42},{ "VOLUNTEERID":111,"CRABID":43},
{ "VOLUNTEERID":111,"CRABID":77},{ "VOLUNTEERID":111,"CRABID":125},{ "VOLUNTEERID":111,"CRABID":173},
{ "VOLUNTEERID":112,"CRABID":10},{ "VOLUNTEERID":112,"CRABID":28},{ "VOLUNTEERID":112,"CRABID":35},
{ "VOLUNTEERID":112,"CRABID":53},{ "VOLUNTEERID":112,"CRABID":68},{ "VOLUNTEERID":112,"CRABID":77},
{ "VOLUNTEERID":112,"CRABID":99},{ "VOLUNTEERID":112,"CRABID":174},{ "VOLUNTEERID":113,"CRABID":23},
{ "VOLUNTEERID":113,"CRABID":49},{ "VOLUNTEERID":113,"CRABID":65},{ "VOLUNTEERID":113,"CRABID":73},
{ "VOLUNTEERID":113,"CRABID":76},{ "VOLUNTEERID":113,"CRABID":98},{ "VOLUNTEERID":113,"CRABID":181},
{ "VOLUNTEERID":114,"CRABID":1},{ "VOLUNTEERID":114,"CRABID":20},{ "VOLUNTEERID":114,"CRABID":32},
{ "VOLUNTEERID":114,"CRABID":38},{ "VOLUNTEERID":114,"CRABID":41},{ "VOLUNTEERID":114,"CRABID":46},
{ "VOLUNTEERID":114,"CRABID":63},{ "VOLUNTEERID":114,"CRABID":182},{ "VOLUNTEERID":114,"CRABID":191},
{ "VOLUNTEERID":115,"CRABID":14},{ "VOLUNTEERID":115,"CRABID":18},{ "VOLUNTEERID":115,"CRABID":47},
{ "VOLUNTEERID":115,"CRABID":57},{ "VOLUNTEERID":115,"CRABID":68},{ "VOLUNTEERID":115,"CRABID":75},
{ "VOLUNTEERID":115,"CRABID":173},{ "VOLUNTEERID":115,"CRABID":268}}]
```

## Queries:

**NOTE:** All the queries are available inside: “[repo root/backend/appService.js](#)”

**List of queries:** (function, location, functionality, **bold if used to demonstrate**)

1. initializeBeachkeepers()
  - a. Line: 141
  - b. Functionality: execute all lines inside the sql script that is used to drop, create tables and insert dummy data. Helps to skip having to manually doing sqlplus Ora\_CWL@stu ... start beachkeepers.sql
2. fetchAllTables()
  - a. Line: 80
  - b. Functionality: fetch all the table names available in the database.  
Used to help the demonstration of the other queries.
3. fetchAllColumns(table)
  - a. Line: 100
  - b. Functionality: fetch all the column names of the given table name.  
Used to assist the Projection functionality.
4. fetchAllTuplesOfColumns(table, columns)
  - a. Line: 121
  - b. Functionality: fetch all the tuples with the selected columns from a specified table.
  - c. **Used to demonstrate *Projection operation*.**
5. fetchAllTuples(tableName)
  - a. Line: 170
  - b. Functionality: fetch all tuples from a specified table.  
Used to help the demonstration of the other queries.
6. insertTrapData(baittype, traptypes, location, age, trapid, phone)
  - a. Line: 524
  - b. Functionality: insert a trap tuple with given attributes into the trap table.
7. insertCrabData(crabid, species, sex, crab\_size, injury, trapid)
  - a. Line: 202
  - b. Functionality: insert a Crab tuple with given attributes into the crab table
  - c. **Used to demonstrate *INSERT operation*.**
8. checkCrabIdExists(crabid)
  - a. Line: 223
  - b. Functionality: check if a crab with the given CrabID exists in the crab table  
Used to assist the INSERT and UPDATE functionalities.
9. updateCrabData(crabid, species, sex, crab\_size, injury, trapid)
  - a. Line: 246
  - b. Functionality: update a Crab tuple with given crabID and other attributes
  - c. **Used to demonstrate *UPDATE operation*.**
10. crabRegularQuery(species, sex, injuries, minSize)
  - a. Line: 271
  - b. Functionality: fetch all crab tuples with specific restrictions
  - c. **Used to demonstrate *Selection operation*.**

11. crabDetailedQuery(species, sex, injuries, minSize, baitType)
  - a. Line: 320
  - b. Functionality: join the crab and trap tables and fetch all tuples with specific restrictions. Shows the crab info and the trap info that caught this crab.
  - c. **Used to demonstrate *Join operation*.**
12. crabSpecialQueryBiggestCatch(species, sex, injuries)
  - a. Line: 379
  - b. Functionality: group the crabs by their species and find the maximum weighted crabs of each species.
  - c. **Used to demonstrate *Aggregation with GROUP BY operation*.**
13. crabSpecialQueryCrabCount(species, sex, injuries, minSize, minCaught)
  - a. Line: 425
  - b. Functionality: join the crab and trap tables, group them by trap location and count how many crabs are caught in each trap location.
  - c. **Used to demonstrate *Aggregation with HAVING operation*.**
14. fetchPositions(shiftDate)
  - a. Line: 487
  - b. Functionality: fetch all the positions assigned to the shift at a given shift date
15. deleteShift(shiftDate)
  - a. Line: 550
  - b. Functionality: delete a shift tuple with specific shift date from the shift table. Will also delete all the position tuples and the registers tuples.
  - c. **Used to demonstrate *DELETE operation*.**
16. insertSupervisor(firstname, lastname, password, phone)
  - a. Line: 577
  - b. Functionality: insert a supervisor tuple with given attributes, executed during the sign up process.
17. insertEmail(firstname, lastname, email, employeeid)
  - a. Line: 604
  - b. Functionality: insert the email tuple into the email tuple that is denormalized from the supervisor schema, executed during the sign up process.
18. insertVolunteer(firstname, lastname, employeeid, password, experience = 0)
  - a. Line: 631
  - b. Functionality: insert the volunteer tuple with given attributes, executed during the sign up process.
19. login(isVolunteer, firstname, lastname, password)
  - a. Line: 659
  - b. Functionality: check if the volunteer or supervisor exists in the database, executed during the login process.
20. getSupervisorIdByName(firstname, lastname)
  - a. Line: 687
  - b. Functionality: get the supervisor id of the given supervisor first and last name.
21. getVolunteerIdByName(firstname, lastname)
  - a. Line: 712
  - b. Functionality: get the supervisor id of the given supervisor first and last name.

22. checkParkVisitorExists(phone)
  - a. Line: 737
  - b. Functionality: check if the park visitor exists in the database, used during inserting interaction tuples.
23. checkCrabberExists(phone)
  - a. Line: 755
  - b. Functionality: check if the crabber exists in the database, used during inserting trap tuples.
24. insertPV(age, firstname, phone)
  - a. Line: 774
  - b. Functionality: insert a park visitor tuple with given attributes, only park visitor inserted allows users to insert crabber, the ISA subclass, with the same primary key.
25. insertCrabber(phone, licenseNumber)
  - a. Line: 800
  - b. Functionality: insert a crabber tuple with given attributes, help creating the ISA relationship.
26. insertInteraction(phone, topic)
  - a. Line: 826
  - b. Functionality: insert a interaction tuple that represent the relationship of volunteer interacts with a park visitor
27. getAvgVolunteerStudiedCrabs()
  - a. Line: 852
  - b. Functionality: group the crabIDs from studies table by volunteerID, and calculate the average count of crabIDs of each volunteer
  - c. **Used to demonstrate *Nested Aggregation operation*.**
28. fetchSpecialTrapIDs()
  - a. Line: 884
  - b. Functionality: Find the traps that caught all kinds of crabs
  - c. **Used to demonstrate *Division operation*.**



### INSERT Operation:

We have created multiple INSERTION queries for different tables, and we will be demonstrating with **inserting crab data**:

- Crab:
  - Function: `insertCrabData(crabid, species, sex, crab_size, injury, trapid)`
  - Line: 202

Demonstration:

- Frontend URL: `"localhost:3000/volunteer/dataUpload"`
- Before:

```
{ "CRABID": 191, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 84, "INJURY": "N", "TRAPID": 11 },  
{ "CRABID": 267, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 93, "INJURY": "N", "TRAPID": 8 },  
{ "CRABID": 268, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 89, "INJURY": "N", "TRAPID": 13 } }
```

We can see that there are no crabs after CrabID = 268, and we will be inserting a new crab into the database with CrabID = 300.

- During:

The screenshot shows a web browser at `localhost:3000/volunteer/dataUpload`. The page has a dark blue header with three tabs: "Dash Board", "Calendar", and "Upload". The "Upload" tab is active. Below the header, there are three light blue panels. The first panel, titled "Crab", has a "Crab ID" field with the value "300", a sub-header "Uploading New Crab", and dropdown menus for "Species" (Green), "Sex" (Female), and "Injury" (No). It also has text input fields for "Size" (70) and "TrapID" (3), and a "Submit" button. The second panel, titled "Trap", has dropdown menus for "Bait Type", "Trap Type", "Location", and "Age Group", and text input fields for "Trap ID", "First Name", "Phone", and "License Number", with a "Submit" button. The third panel, titled "Interaction", has text input fields for "First Name", "Phone", "Topic", and "Age Group", and a "Submit" button.

In the data upload page, we will have three sections, uploading crab, trap and interaction. For this demonstration, we will only enter data into the Crab section.

This is a close-up of the "Crab" section from the previous screenshot. It shows the "Crab ID" field with the value "300". Below it is the sub-header "Uploading New Crab". There are three dropdown menus: "Species" with "Green" selected, "Sex" with "Female" selected, and "Injury" with "No" selected. Below these are two text input fields: "Size" with the value "70" and "TrapID" with the value "3". At the bottom is a dark blue "Submit" button.

Now, we have entered all the fields in the report form, by clicking submit, our frontend will send a POST request to the backend, if we successfully added the data, we will receive a notification indicating submission successful (see below).



## University of British Columbia, Vancouver

### Department of Computer Science

Crab	Trap
Crab ID: <input type="text" value="300"/>	Bait Type: <input type="text" value="Select"/>
Crab Exists In System!	Trap Type: <input type="text" value="Select"/>
Species: <input type="text" value="Green"/>	Location: <input type="text" value="Select"/>
Sex: <input type="text" value="Female"/>	Age Group: <input type="text" value="Select"/>
Injury: <input type="text" value="No"/>	Trap ID: <input type="text" value="Successfully submitted!"/>
Size: <input type="text" value="70"/>	First Name: <input type="text"/>
TrapID: <input type="text" value="3"/>	Phone: <input type="text"/>
<input type="button" value="Update"/>	License Number: <input type="text"/>

Notice that in the meantime, the indication saying “uploading new crab” now transforms into a red “crab exists in system”, and the submit button switches to update. We will explain updates later.

- After:

```
{ "CRABID": 267, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 93, "INJURY": "N", "TRAPID": 8 },  
{ "CRABID": 300, "SPECIES": "Green", "SEX": "F", "CRAB_SIZE": 70, "INJURY": "N", "TRAPID": 3 } }
```

When we refresh the data, we can see that we got a new crab that has all the fields we have entered.

### DELETE Operation:

We only have one delete operation used to **delete shifts**:

- Shift
  - Function: `deleteShift(shiftDate)`
  - Line: 550

Demonstration:

- Frontend url: "`localhost:3000/supervisor/calendar`"
- Before:

```
{
  "SHIFT_DATE": "2023-06-24T07:00:00.000Z",
  "LOWTIDE": "10:34:23",
  "HIGHTIDE": "21:59:00"
},
{
  "SHIFT_DATE": "2023-07-09T07:00:00.000Z",
  "LOWTIDE": "10:39:48",
  "HIGHTIDE": "22:11:08"
},
{
  "SHIFT_DATE": "2024-06-30T07:00:00.000Z",
  "LOWTIDE": "10:52:28",
  "HIGHTIDE": "23:18:47"
},
{
  "SHIFT_DATE": "2024-07-01T07:00:00.000Z",
  "LOWTIDE": "11:44:53",
  "HIGHTIDE": "24:42:12"
},
{
  "SHIFT_DATE": "2024-07-13T07:00:00.000Z",
  "LOWTIDE": "12:26:27",
  "HIGHTIDE": "00:33:06"
}
```

We have got five shifts in our database and they have been displayed both in "`volunteer/calendar`" and "`supervisor/calendar`". Since the dummy shifts are in July, please navigate the calendar to the correct month to see them displayed.

- During:

The screenshot shows a web application interface for a supervisor's calendar. The main area displays a calendar for July 2024, with a 'Today' button and navigation links (Back, Next). The calendar shows shifts for the month, with specific shifts highlighted: 'Shift at wharf' on Sunday, June 30, and 'Shift at picnic area' on Monday, July 1. The sidebar on the right contains a 'Roles' section with three checkboxes (Role 1, Role 2, Role 3), a 'Location' dropdown menu set to 'Location 1', a 'Date' field, and a 'Create Shift' button.

In the above picture, we can see that there are two July shifts and one June shift displayed on the calendar. As the supervisor, we can click on the shift block, and the "create shift" button on the bottom right corner will turn into a "delete shift" button.

12	Shift at wharf	13
19		20
26		27
02		03

Location:

Location 1

Date:

2024-07-13

Delete Shift

If we click on “Delete Shift” and refresh the page, we will see the shift is removed from the calendar.

Today Back Next

July 2024

Month Week Day Agenda

Sun	Mon	Tue	Wed	Thu	Fri	Sat
30	01	02	03	04	05	06
Shift at wharf	Shift at picnic area					
07	08	09	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	01	02	03

Roles:

☐ Role 1

☐ Role 2

☐ Role 3

Location:

Location 1

Date:

Create Shift

- After:  
In the meantime, the data in “/tables/fetch?tableName=shift” will be removed too.

```
{
  "SHIFT_DATE": "2023-06-24T07:00:00.000Z",
  "LOWTIDE": "10:34:23",
  "HIGHTIDE": "21:59:00"
},
{
  "SHIFT_DATE": "2023-07-09T07:00:00.000Z",
  "LOWTIDE": "10:39:48",
  "HIGHTIDE": "22:11:08"
},
{
  "SHIFT_DATE": "2024-06-30T07:00:00.000Z",
  "LOWTIDE": "10:52:28",
  "HIGHTIDE": "23:18:47"
},
{
  "SHIFT_DATE": "2024-07-01T07:00:00.000Z",
  "LOWTIDE": "11:44:53",
  "HIGHTIDE": "24:42:12"
}
```

### UPDATE Operation:

We have introduced our only UPDATE operation previously inside INSERT crab, where if the crabID exists, we will prompt the user that the data will **update the existing crab**.

- Crab:
  - Function: `updateCrabData(crabid, species, sex, crab_size, injury, trapid)`
  - Line: 246

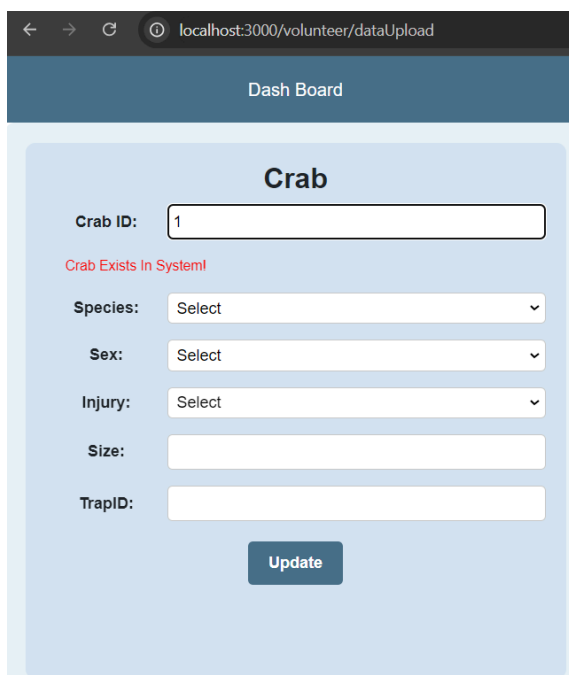
### Demonstration:

- Frontend URL: `localhost:3000/volunteer/dataUpload`
- Before:

```
{ "success": true, "rows": [ { "CRABID": 1, "SPECIES": "Red rock", "SEX": "M", "CRAB_SIZE": 108, "INJURY": "Y", "TRAPID": 5 },  
  { "CRABID": 2, "SPECIES": "Graceful", "SEX": "F", "CRAB_SIZE": 59, "INJURY": "Y", "TRAPID": 2 },  
  { "CRABID": 3, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 103, "INJURY": "N", "TRAPID": 3 },  
  { "CRABID": 4, "SPECIES": "Graceful", "SEX": "M", "CRAB_SIZE": 94, "INJURY": "N", "TRAPID": 3 } ] }
```

We have a set of crab data starting from crabID = 1, let's modify the first crab and change all of its attributes.

- During:



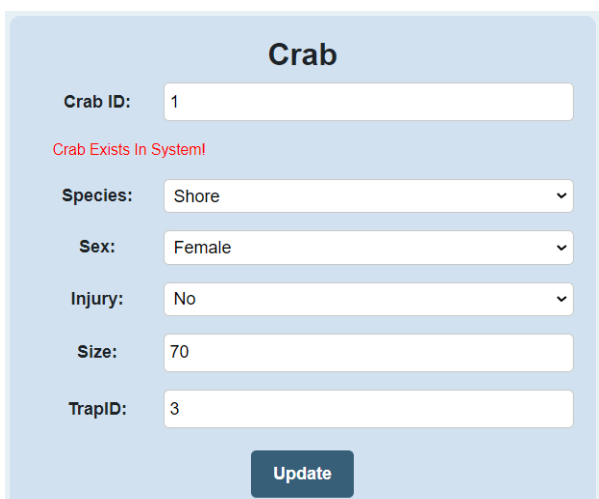
In the upload page, we have just entered 1 into the Crab ID section. After communicating with the backend, we are notified that this crab exists in our database; thus, the "Submit" button now changes to "Update".

The original crab has the attributes of:

"SPECIES": "Red rock",  
"SEX": "M",  
"CRAB\_SIZE": 108,  
"INJURY": "Y",  
"TRAPID": 5

Lets change them to:

"SPECIES": "Shore",  
"SEX": "F",  
"CRAB\_SIZE": 70,  
"INJURY": "N",  
"TRAPID": 3



After clicking "Update", the frontend will send a PUT request to the correct backend endpoint and trigger the updateCrabData function.

# University of British Columbia, Vancouver

## Department of Computer Science

### Crab

Crab ID:

Crab Exists In System!

Species:

Sex:

Injury:

Size:

TrapID:

### Trap

Bait Type:

Trap Type:

Location:

Age Group:

Trap ID:

First Name:

Phone:

License Number:

Similar to submit, a green notification will be displayed at the center of the screen if the update is successful.

- After:

In the meantime, the data in `"/tables/fetch?tableName=crab"` will be modified too.

```
{ "success": true, "rows": [ { "CRABID": 1, "SPECIES": "Shore", "SEX": "F", "CRAB_SIZE": 70, "INJURY": "N", "TRAPID": 3 },  
  { "CRABID": 2, "SPECIES": "Graceful", "SEX": "F", "CRAB_SIZE": 59, "INJURY": "Y", "TRAPID": 2 },  
  { "CRABID": 3, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 103, "INJURY": "N", "TRAPID": 3 },  
  { "CRABID": 4, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 103, "INJURY": "N", "TRAPID": 3 } ] }
```

### Selection Operation:

We have created multiple SELECTION queries for different purposes, for the demonstration we will be using:

- Querying crab data based on restrictions over its attributes
  - Function: `crabRegularQuery(species, sex, injuries, minSize)`
  - Lline: 271

**Users can manually specify specific species, sex, injuries, and minimum size on the frontend GUI, and the Query will select the crabs with correct specification.** These data will then be displayed in the frontend.

Demonstration:

- Frontend URL: `localhost:3000/public/crab-insight`
- Before:

```
{
  "success": true,
  "rows": [
    {
      "CRABID": 1,
      "SPECIES": "Shore",
      "SEX": "F",
      "CRAB_SIZE": 70,
      "INJURY": "N",
      "TRAPID": 3
    },
    {
      "CRABID": 2,
      "SPECIES": "Graceful",
      "SEX": "F",
      "CRAB_SIZE": 59,
      "INJURY": "Y",
      "TRAPID": 2
    },
    {
      "CRABID": 3,
      "SPECIES": "Dungeness",
      "SEX": "F",
      "CRAB_SIZE": 103,
      "INJURY": "N",
      "TRAPID": 3
    },
    {
      "CRABID": 4,
      "SPECIES": "Graceful",
      "SEX": "M",
      "CRAB_SIZE": 94,
      "INJURY": "N",
      "TRAPID": 3
    },
    {
      "CRABID": 5,
      "SPECIES": "Dungeness",
      "SEX": "F",
      "CRAB_SIZE": 87,
      "INJURY": "Y",
      "TRAPID": 4
    },
    {
      "CRABID": 6,
      "SPECIES": "Dungeness",
      "SEX": "F",
      "CRAB_SIZE": 81,
      "INJURY": "N",
      "TRAPID": 1
    },
    {
      "CRABID": 7,
      "SPECIES": "Dungeness",
      "SEX": "F",
      "CRAB_SIZE": 100,
      "INJURY": "N",
      "TRAPID": 1
    },
    {
      "CRABID": 8,
      "SPECIES": "Dungeness",
      "SEX": "M",
      "CRAB_SIZE": 78,
      "INJURY": "N",
      "TRAPID": 1
    },
    {
      "CRABID": 9,
      "SPECIES": "Graceful",
      "SEX": "F",
      "CRAB_SIZE": 70,
      "INJURY": "N",
      "TRAPID": 1
    },
    {
      "CRABID": 10,
      "SPECIES": "Dungeness",
      "SEX": "M",
      "CRAB_SIZE": 83,
      "INJURY": "N",
      "TRAPID": 1
    },
    {
      "CRABID": 11,
      "SPECIES": "Dungeness",
      "SEX": "F",
      "CRAB_SIZE": 77,
      "INJURY": "N",
      "TRAPID": 1
    },
    {
      "CRABID": 12,
      "SPECIES": "Graceful",
      "SEX": "F",
      "CRAB_SIZE": 77,
      "INJURY": "N",
      "TRAPID": 1
    },
    {
      "CRABID": 13,
      "SPECIES": "Dungeness",
      "SEX": "F",
      "CRAB_SIZE": 71,
      "INJURY": "N",
      "TRAPID": 1
    },
    {
      "CRABID": 14,
      "SPECIES": "Graceful",
      "SEX": "M",
      "CRAB_SIZE": 77,
      "INJURY": "N",
      "TRAPID": 1
    },
    {
      "CRABID": 15,
      "SPECIES": "Dungeness",
      "SEX": "F",
      "CRAB_SIZE": 70,
      "INJURY": "N",
      "TRAPID": 1
    },
    {
      "CRABID": 16,
      "SPECIES": "Graceful",
      "SEX": "F",
      "CRAB_SIZE": 57,
      "INJURY": "N",
      "TRAPID": 1
    },
    {
      "CRABID": 17,
      "SPECIES": "Graceful",
      "SEX": "F",
      "CRAB_SIZE": 73,
      "INJURY": "Y",
      "TRAPID": 1
    }
  ]
}
```

We can see that we have populated our database with various many diverse examples, such as different species, sex, size and injury.

**NOTE:** The first crab data has been modified due to the Update operation

Similarly, we can see these data at `public/crab-insight`:

The screenshot shows a web browser window with the URL `localhost:3000/public/crab-insight`. The application has a dark blue header with the text "General" on the left and "Crabs!!!" on the right. Below the header is a light blue form area. The form contains four dropdown menus arranged in a 2x2 grid: "Species" (set to "Unspecified"), "Sex" (set to "Unspecified"), "Injuries" (set to "Unspecified"), and "Minimum Size" (empty). Below these is a "Query Mode" dropdown set to "Regular". At the bottom of the form is a blue "Submit" button.

This page allows users to specify what data they wish to query. If nothing is specified, the query will act normally as selecting all tuples from the crab table.

Species

Unspecified

Sex

Unspecified

Injuries

Unspecified

Minimum Size

Query Mode

Regular

Submit

0.

CrabID: 1

Species: Shore

Sex: F

CrabSize: 70

Injury: N

TrapID: 3

-----

1.

CrabID: 2

Species: Graceful

Sex: F

CrabSize: 59

Injury: Y

TrapID: 2

-----

After clicking submit, we can see that the data are displayed in the order of crabID, and the first crab is the one we have updated previously.

- During:

Now we will perform selection with parameters specified.

Species

Red Rock

Sex

Female

Injuries

No Injuries

Minimum Size

90

Query Mode

Regular

Here, we will look for Red Rock crabs that are female, no injuries, and at least 90 mm wide. After clicking submit, we will see the data displayed as follows:

- After:

Submit

0.

CrabID: 36

Species: Red rock

Sex: F

CrabSize: 103

Injury: N

TrapID: 1

-----

1.

CrabID: 49

Species: Red rock

Sex: F

CrabSize: 116

Injury: N

TrapID: 9

-----

As we can see, there are only two crabs that satisfy the specification. This can also be proven by manually searching up in all the tuples.

```
{ "CRABID": 25, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 67, "INJURY": "N", "TRAPID": 2 },  
{ "CRABID": 26, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 78, "INJURY": "N", "TRAPID": 6 },  
{ "CRABID": 27, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 67, "INJURY": "N", "TRAPID": 2 },  
{ "CRABID": 28, "SPECIES": "Graceful", "SEX": "F", "CRAB_SIZE": 78, "INJURY": "N", "TRAPID": 6 },  
{ "CRABID": 29, "SPECIES": "Red rock", "SEX": "F", "CRAB_SIZE": 85, "INJURY": "N", "TRAPID": 6 },  
{ "CRABID": 30, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 97, "INJURY": "N", "TRAPID": 6 },  
{ "CRABID": 31, "SPECIES": "Red rock", "SEX": "M", "CRAB_SIZE": 81, "INJURY": "N", "TRAPID": 5 },  
{ "CRABID": 32, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 80, "INJURY": "N", "TRAPID": 5 },  
{ "CRABID": 33, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 74, "INJURY": "N", "TRAPID": 5 },  
{ "CRABID": 34, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 81, "INJURY": "N", "TRAPID": 5 },  
{ "CRABID": 35, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 76, "INJURY": "Y", "TRAPID": 1 },  
{ "CRABID": 36, "SPECIES": "Red rock", "SEX": "F", "CRAB_SIZE": 103, "INJURY": "N", "TRAPID": 1 },  
{ "CRABID": 37, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 78, "INJURY": "N", "TRAPID": 1 },  
{ "CRABID": 38, "SPECIES": "Dungeness", "SEX": "M", "CRAB_SIZE": 81, "INJURY": "N", "TRAPID": 1 },  
{ "CRABID": 39, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 87, "INJURY": "N", "TRAPID": 1 },  
{ "CRABID": 40, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 80, "INJURY": "N", "TRAPID": 1 },  
{ "CRABID": 41, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 80, "INJURY": "N", "TRAPID": 1 },  
{ "CRABID": 42, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 85, "INJURY": "N", "TRAPID": 1 },  
{ "CRABID": 43, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 91, "INJURY": "N", "TRAPID": 10 },  
{ "CRABID": 44, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 91, "INJURY": "N", "TRAPID": 10 },  
{ "CRABID": 45, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 95, "INJURY": "Y", "TRAPID": 10 },  
{ "CRABID": 46, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 91, "INJURY": "N", "TRAPID": 10 },  
{ "CRABID": 47, "SPECIES": "Graceful", "SEX": "F", "CRAB_SIZE": 99, "INJURY": "N", "TRAPID": 10 },  
{ "CRABID": 48, "SPECIES": "Dungeness", "SEX": "F", "CRAB_SIZE": 83, "INJURY": "N", "TRAPID": 10 },  
{ "CRABID": 49, "SPECIES": "Red rock", "SEX": "F", "CRAB_SIZE": 116, "INJURY": "N", "TRAPID": 9 },
```

Notice that although we found three female red rock crabs, one of them is only 85 mm wide, which is smaller than our specified crab size. Thus will not be selected from the table.



### Projection Operation:

The projection operation is available inside

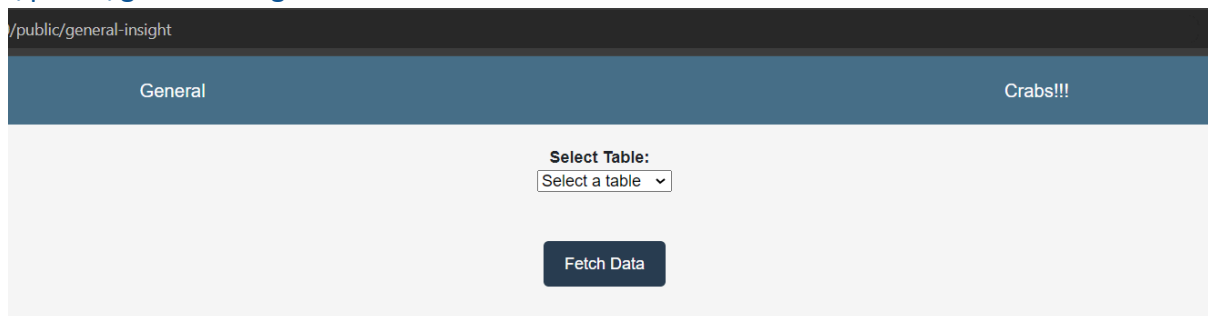
- projecting specified columns of a given table and **fetch all tuples**
  - Function: `fetchAllTuplesOfColumns(table, columns)`
  - Line: 121

Demonstration:

- Frontend URL: `"localhost:3000/public/general-insight"`
- Before:

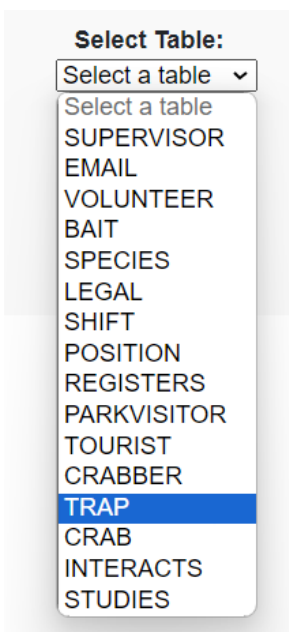
```
{ "success": true, "rows": [{ "CRABID": 1, "SPECIES": "Shore", "SEX": "F", "CRAB_SIZE": 70, "INJURY": "N", "TRAPID": 3 },  
  { "CRABID": 2, "SPECIES": "Graceful", "SEX": "F", "CRAB_SIZE": 59, "INJURY": "Y", "TRAPID": 2 },  
  
  { "success": true, "rows":  
    [ { "TRAPID": 1, "BAITTYPE": "Chicken", "TRAPTYPE": "Circle", "TRAP_LOCATION": "East", "PHONE": "778-284-9134",  
      { "TRAPID": 2, "BAITTYPE": "Turkey", "TRAPTYPE": "Clamshell", "TRAP_LOCATION": "Main", "PHONE": "604-038-1435",  
      { "TRAPID": 3, "BAITTYPE": "Turkey", "TRAPTYPE": "Clamshell", "TRAP_LOCATION": "North", "PHONE": "604-678-4319",
```

The above are two tables with four or more attributes: trap, and crab. For this demonstration, we will go with the trap table. Lets first head to the frontend page, `"/public/general-insight"`:



In this page, users may freely select any table that is available inside our database and specify the columns they want to see.

- During:  
For example, if we want to see only the traptype and trap\_location of the traps, we can do the following.



1. Select the desired table from the drop down menu.

2. Select the columns present in the database.

Select Table:  
TRAP

Select Columns:

☐ TRAPID

☐ BAITTYPE

☒ TRAPTYPE

☒ TRAP\_LOCATION

☐ PHONE

**NOTE:** the table options and the column options in the drop down menu and the checkboxes are handled by fetch operations that can look for all tables and all columns. Thus we do not need to worry about changing the schema.

- After:  
We have now specified all the columns we wish to see; we can then click “Fetch Data” to see them displayed.

Select Columns:

☐ TRAPID

☐ BAITTYPE

☒ TRAPTYPE

☒ TRAP\_LOCATION

☐ PHONE

Fetch Data

TRAPTYPE	TRAP_LOCATION
Circle	East
Clamshell	Main
Clamshell	North
Box	South
Clamshell	South

We can see that only the traptype and the trap\_location are displayed on the page. The same will happen if we select one more column, baittype.

Select Columns:

☐ TRAPID

☒ BAITTYPE

☒ TRAPTYPE

☒ TRAP\_LOCATION

☐ PHONE

Fetch Data

TRAPTYPE	TRAP_LOCATION	BAITTYPE
Circle	East	Chicken
Clamshell	Main	Turkey
Clamshell	North	Turkey
Box	South	Pork

Clearly, only the three specified columns are displayed.

```
{
  "success": true,
  "rows": [
    {
      "TRAPID": 1,
      "BAITTYPE": "Chicken",
      "TRAPTYPE": "Circle",
      "TRAP_LOCATION": "East",
      "PHONE": "778-284-9134"
    },
    {
      "TRAPID": 2,
      "BAITTYPE": "Turkey",
      "TRAPTYPE": "Clamshell",
      "TRAP_LOCATION": "Main",
      "PHONE": "604-038-1435"
    },
    {
      "TRAPID": 3,
      "BAITTYPE": "Turkey",
      "TRAPTYPE": "Clamshell",
      "TRAP_LOCATION": "North",
      "PHONE": "604-678-4319"
    },
    {
      "TRAPID": 4,
      "BAITTYPE": "Pork",
      "TRAPTYPE": "Box",
      "TRAP_LOCATION": "South",
      "PHONE": "642-174-8903"
    },
    {
      "TRAPID": 5,
      "BAITTYPE": "Chicken",
      "TRAPTYPE": "Clamshell",
      "TRAP_LOCATION": "South",
      "PHONE": "642-174-8903"
    }
  ]
}
```

We can also see that the data being displayed matches exactly with the order of querying all tuples from the trap table with no restriction.

### Join Operation:

The join operation is present over:

- Fetch all crabs and trap that caught these crabs
  - Function: `crabDetailedQuery(species, sex, injuries, minSize, baitType)`
  - Line: 320

This operation **joins the crab table and the trap table**, and displays the crab data and the trap that caught this crab. Similar to the previous Selection Operation, users may specify the specific crab species, sex, injuries, and minimum size to filter out the unwanted crabs. Moreover, users may also specify the baitType to apply a more detailed restriction on the traps.

### Demonstration:

- Frontend URL: `“localhost:3000/public/crab-insight”`
- Before:

```
{ "success": true, "rows":  
[{"TRAPID":1,"BAITTYPE":"Chicken","TRAPTYPE":"Circle","TRAP_LOCATION":"East","PHONE":"778-284-9134"},  
{"TRAPID":2,"BAITTYPE":"Turkey","TRAPTYPE":"Clamshell","TRAP_LOCATION":"Main","PHONE":"604-038-1435"},  
{"TRAPID":3,"BAITTYPE":"Turkey","TRAPTYPE":"Clamshell","TRAP_LOCATION":"North","PHONE":"604-678-4319"},  
{"TRAPID":4,"BAITTYPE":"Pork","TRAPTYPE":"Box","TRAP_LOCATION":"South","PHONE":"642-174-8903"},  
{"TRAPID":5,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"South","PHONE":"642-174-8903"},  
{"TRAPID":6,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"East","PHONE":"256-397-9103"},  
{"TRAPID":10,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"Main","PHONE":"256-397-9103"},  
{"TRAPID":12,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"North","PHONE":"185-274-5245"},  
{"TRAPID":14,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"East","PHONE":"374-939-2754"},  
{"TRAPID":13,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"North","PHONE":"374-939-2754"},  
{"TRAPID":15,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"South","PHONE":"185-274-5245"},  
{"TRAPID":17,"BAITTYPE":"Chicken","TRAPTYPE":"Clamshell","TRAP_LOCATION":"East","PHONE":"245-024-5202"}]
```

We can see that most traps use chicken as their baittype, only a few use turkey. Let's try to find a crab that is caught by a turkey bait trap. Let's navigate to our frontend, `“/public/crab-insight”`:

General

Crabs!!!

Species

Sex

Unspecified

Unspecified

Injuries

Minimum Size

Unspecified

Query Mode

Regular

Submit

- During:

Our page is the same as the one used to demonstrate select operations; however, specifying the Query Mode as “Detailed”, we can enable the join functionality.

Query Mode

Bait Type

Detailed

Unspecified

As we can see, if we set the Query Mode drop down menu to detailed, a new bait type drop down menu will appear, and this is where we can set restrictions to the trap table. Let’s firstly see the join operation with out specific bait type:

<b>Species</b>	<b>Sex</b>
<div>Graceful</div>	<div>Male</div>
<b>Injuries</b>	<b>Minimum Size</b>
<div>No Injuries</div>	<div>85</div>
<b>Query Mode</b>	<b>Bait Type</b>
<div>Detailed</div>	<div>Unspecified</div>

Here, we are looking for male Graceful crabs with no injuries and at least 85 mm. Let’s see our query result:

- After:

0.
CrabID: 4
Species: Graceful
Sex: M
CrabSize: 94
Injury: N
TrapID: 3
BaitType: Turkey
TrapType: Clamshell
TrapLocation: North
-----
1.
CrabID: 50
Species: Graceful
Sex: M
CrabSize: 85
Injury: N
TrapID: 9
BaitType: Chicken
TrapType: Clamshell
TrapLocation: Main
-----
2.
CrabID: 76
Species: Graceful
Sex: M
CrabSize: 95
Injury: N
TrapID: 16
BaitType: Chicken
TrapType: Clamshell

Seems like only three crabs satisfy the specifications, and only one of them is caught by the turkey bait trap.

Intuitively, if we specify the bait type to turkey, only one tuple should be displayed.

<b>Species</b> Graceful ▾	<b>Sex</b> Male ▾
<b>Injuries</b> No Injuries ▾	<b>Minimum Size</b> 85
<b>Query Mode</b> Detailed ▾	<b>Bait Type</b> Turkey ▾
<b>Submit</b>	
<p>0. <b>CrabID:</b> 4 <b>Species:</b> Graceful <b>Sex:</b> M <b>CrabSize:</b> 94 <b>Injury:</b> N <b>TrapID:</b> 3 <b>BaitType:</b> Turkey <b>TrapType:</b> Clamshell <b>TrapLocation:</b> North</p> <p>-----</p>	

And that is what happens. Only one crab with our specifications is caught by a turkey bait trap.

### Aggregation with GROUP BY Operation:

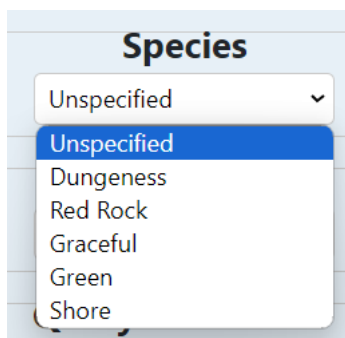
We will demonstrate **aggregation** on the same frontend page as the join and selection operation, and the function in the backend is located at:

- Group the crabs by species and find the biggest ones of each type.
  - Function: crabSpecialQueryBiggestCatch(species, sex, injuries)
  - Line: 379

Demonstration:

- Frontend URL: "[localhost:3000/public/crab-insight](http://localhost:3000/public/crab-insight)"
- Before:

Let's first navigate to the frontend page, "[public/crab-insight](http://localhost:3000/public/crab-insight)"

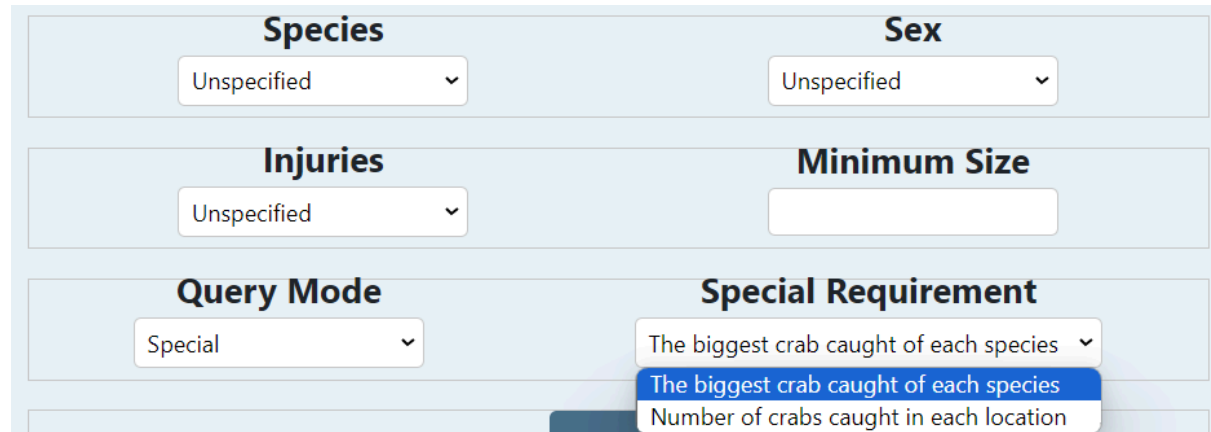


We can see that there are five different types of crabs available in the database.

**NOTE:** There were no Green and Shore crabs in our sql script; however, we have added a Green crab through the INSERT operation and changed a crab's species to Shore through the UPDATE operation during demonstration.

- During:

Let's now change the Query Mode to "Special"



A new drop down menu, "Special Requirement" will appear. By default, we will be looking for "the biggest crab caught of each species". The other one, "number of crabs caught in each location" is used for "Aggregation with Having", which will be demonstrated in the next section.

- After:

As we can see below, only five tuples, one of each kind of crab species, are selected from the database. The BiggestCatch shows the maximum weight of each kind of crab, the values for Green and Shore crabs match exactly as the Green and Shore crabs we have just inserted and updated.

Query Mode

Special

Special Requirement

The biggest crab caught of each species

Submit

0.

Species: Green

BiggestCatch: 70

1.

Species: Shore

BiggestCatch: 70

2.

Species: Dungeness

BiggestCatch: 152

3.

Species: Graceful

BiggestCatch: 99

4.

Species: Red rock

BiggestCatch: 116

We can also verify the other species by going to “Regular” query mode used in the selection operation and specifying the minimum size to the BiggestCatch. For example:

Species

Dungeness

Sex

Unspecified

Injuries

Unspecified

Minimum Size

152

Query Mode

Regular

Submit

0.

CrabID: 59

Species: Dungeness

Sex: F

CrabSize: 152

Injury: N

TrapID: 10

We can see that only one Dungeness crab is selected from the database with the minimum size at 152 mm, which proves that our biggest catch operation is correct.

### Aggregation with HAVING Operation:

We will demonstrate aggregation on the same frontend page as the join, selection, and aggregation with group by operations, and the function in the backend is located at:

- Find the number of crabs caught in each location
  - Function: `crabSpecialQueryCrabCount(species, sex, injuries, minSize, minCaught)`
  - Line: 425

Demonstration:

- Frontend URL: `localhost:3000/public/crab-insight`
- Before:

```
[{"TRAPID":1,"BAITTYPE":"Chicken","TRAPTYPE":"Circle","TRAP_LOCATION":"East","PHONE":"778-284-9134"}, {"TRAPID":2,"BAITTYPE":"Turkey","TRAPTYPE":"Clamshell","TRAP_LOCATION":"Main","PHONE":"604-038-1435"}, {"TRAPID":3,"BAITTYPE":"Turkey","TRAPTYPE":"Clamshell","TRAP_LOCATION":"North","PHONE":"604-678-4319"}, {"TRAPID":4,"BAITTYPE":"Pork","TRAPTYPE":"Box","TRAP_LOCATION":"South","PHONE":"642-174-8903"}]
```

There are four trap locations in our database: "East", "Main", "North", and "South".

- During:

Let's navigate to the frontend page: `/public/crab-insight`, change the query mode to "Special", and specify the special requirement as "number of crabs caught in each location".

The screenshot shows a form with the following fields and values:

Species	Sex
Unspecified	Unspecified

Injuries	Minimum Size
Unspecified	

Query Mode	Special Requirement	Minimum Caught
Special	Number of crabs caught in each location	

We can see that a new number input box, minimum caught, appears; this will allow users to filter out locations that caught crabs under this specified number. Let's first query without minimum caught specification.

- After:

The screenshot shows the results of a query for the number of crabs caught in each location. The results are displayed in a list format:

Location	CrabCount
North	17
East	34
South	19
Main	31



The traps together have caught 17 crabs in the North shore, which is the lowest in all locations. If we set the minimum catch number to 18, supposedly the north shore will be filtered out.

Special	Number of crabs caught in each location	18
<b>Submit</b>		
0. <b>Location:</b> East <b>CrabCount:</b> 34		
-----		
1. <b>Location:</b> South <b>CrabCount:</b> 19		
-----		
2. <b>Location:</b> Main <b>CrabCount:</b> 31		
-----		

**NOTE:** Both Aggregation with Group By and Aggregation with Having can be further combined with the regular specifications. E.g, users can see what are the largest male crabs with injuries, or how many graceful crabs of at least 90 mm are caught in each location.

- Largest size for the male crabs with injuries:

Unspecified	Male
<b>Injuries</b>	<b>Minimum Size</b>
Injuries	
<b>Query Mode</b>	<b>Special Requirement</b>
Special	The biggest crab caught of each species
<b>Submit</b>	
0. <b>Species:</b> Dungeness <b>BiggestCatch:</b> 97	
-----	
1. <b>Species:</b> Graceful <b>BiggestCatch:</b> 80	
-----	

- Number of graceful crabs of at least 70 mm caught in each location:

Graceful	Unspecified	
<b>Injuries</b>	<b>Minimum Size</b>	
Unspecified	90	
<b>Query Mode</b>	<b>Special Requirement</b>	<b>Minimum Caught</b>
Special	Number of crabs caught in each location	
<b>Submit</b>		
0. <b>Location:</b> North <b>CrabCount:</b> 1		
-----		
1. <b>Location:</b> Main <b>CrabCount:</b> 2		
-----		

### Nested Aggregation with GROUP BY Operation:

We will demonstrate **aggregation** on another page, the same as the one we used to demonstrate projection: “localhost:3000/public/general-insight”

The function in the backend is located at:

- group the crabIDs from studies table by volunteerID, and calculate the average count of crabIDs of each volunteer
  - Function: getAvgVolunteerStudiedCrabs()
  - Line: 852

Demonstration:

- **Note:** We have re-initiated the database for better demonstration purposes.
- Frontend URL: /public/general-insight
- Before:

Let’s first see how many “studies” tuple exists in our database:

```
437 insert into studies values(114, 1);
438 insert into studies values(103, 1);
439 insert into studies values(104, 2);
...
570 insert into studies values(114, 191);
571 insert into studies values(109, 267);
572 insert into studies values(115, 268);
```

```
198 insert into volunteer values(100, 'Elayne', 'Lu', '123', 9, 1);
199 insert into volunteer values(101, 'Lisa', 'Kooz', '123', 14, 5);
200 insert into volunteer values(102, 'Cheryl', 'Cheng', '123', 10, 5);
201 insert into volunteer values(103, 'Calum', 'Lederat', '123', 4, 2);
202 insert into volunteer values(104, 'Eric', 'Dunce', '123', 5, 3);
203 insert into volunteer values(105, 'Jordin', 'Thumper', '123', 5, 3);
204 insert into volunteer values(106, 'Baley', 'Johnston', '123', 7, 3);
205 insert into volunteer values(107, 'Danny', 'Wang', '123', 6, 4);
206 insert into volunteer values(108, 'Kevin', 'Xu', '123', 8, 2);
207 insert into volunteer values(109, 'Brute', 'Yang', '123', 11, 1);
208 insert into volunteer values(110, 'Jazzy', 'Singh', '123', 4, 1);
209 insert into volunteer values(111, 'Ani', 'Rube', '123', 2, 2);
210 insert into volunteer values(112, 'Dora', 'Rose', '123', 1, 5);
211 insert into volunteer values(113, 'Kirsty', 'Fluver', '123', 0, 5);
212 insert into volunteer values(114, 'Orla', 'Sharma', '123', 12, 4);
213 insert into volunteer values(115, 'Uma', 'Smith', '123', 3, 4);
```

There are roughly 136 tuples here, which means the volunteers have studied 136 crabs.

And we have 16 volunteers in our database.

- During:

Now let’s go to the frontend page, and select the table to view as Volunteer.

The screenshot shows a web interface with a dark blue header containing 'General' and 'Crabs!!!' tabs. Below the header, there's a 'Select Table:' dropdown menu with 'VOLUNTEER' selected. Underneath, a 'Select Columns:' section lists several columns: TRAPID, BAITYPE, TRAPTYPE, TRAP\_LOCATION, and PHONE, each with a checked checkbox. At the bottom of this section is a 'Fetch Data' button.

After clicking “Fetch Data”, we can see that all the volunteer tuples are being fetched out, and if we scroll to the bottom of the page we can see a line displaying the average number of crabs studied by each volunteer.

- After:

Danny	Wang	123	0
Kevin	Xu	123	8
Brute	Yang	123	11
.Jazzz	Sinh	123	4
Average number of crabs studied by all volunteers: <b>8.5</b>			

We can see that on average each volunteer studied 8.5 crabs. This is true by  $136 \text{ studied} / 16 \text{ volunteers} = 8.5 \text{ crabs studied by each volunteer}$ .

### Division Operation:

We will demonstrate **division** on the same page as the one we used to demonstrate projection and Nested Aggregation with GROUP BY: “localhost:3000/public/general-insight”

The function in the backend is located at:

- Find the traps that caught all kinds of crabs
  - Function: fetchSpecialTrapIDs()
  - Line: 884

Demonstration:

- Frontend URL: /public/general-insight
- Before:

Right now we have re-initialized our database. By default, we have only populated our database with three out of five different kinds of crab species. If we head to the frontend page, and select to display the tuples of traps, we will receive an error:

The screenshot shows the 'General' tab of the application. A dark error message box is displayed at the top right, stating 'localhost:3000 says' and 'Unable to fetch special traps that caught all five crabs', with an 'OK' button. Below the error, a 'Select Columns:' dialog is open, showing a list of columns with checkboxes: 'Select All', 'TRAPID', 'BAITTYPE', 'TRAPTYPE', 'TRAP\_LOCATION', and 'PHONE'. All checkboxes are currently checked.

This is because right now no traps have caught all five kinds of crabs. Let's go to the volunteer upload page and update the first five crabs with ID (1,2,3,4,5), each to a different species and assign the Trap ID = 1.

The image shows two side-by-side screenshots of the 'Crab' update form. The left form is for Crab ID 1, with 'Dungeness' selected for Species, 'Select' for Sex and Injury, an empty field for Size, and '1' for TrapID. The right form is for Crab ID 2, with 'Red Rock' selected for Species, 'Select' for Sex and Injury, an empty field for Size, and '1' for TrapID. Both forms have an 'Update' button at the bottom.

And we will perform three more update operations similar to the above pictures.

```
{
  "success": true,
  "rows": [
    {
      "CRABID": 1,
      "SPECIES": "Dungeness",
      "SEX": null,
      "CRAB_SIZE": null,
      "INJURY": null,
      "TRAPID": 1
    },
    {
      "CRABID": 2,
      "SPECIES": "Red rock",
      "SEX": null,
      "CRAB_SIZE": null,
      "INJURY": null,
      "TRAPID": 1
    },
    {
      "CRABID": 3,
      "SPECIES": "Graceful",
      "SEX": null,
      "CRAB_SIZE": null,
      "INJURY": null,
      "TRAPID": 1
    },
    {
      "CRABID": 4,
      "SPECIES": "Green",
      "SEX": null,
      "CRAB_SIZE": null,
      "INJURY": null,
      "TRAPID": 1
    },
    {
      "CRABID": 5,
      "SPECIES": "Shore",
      "SEX": null,
      "CRAB_SIZE": null,
      "INJURY": null,
      "TRAPID": 1
    }
  ]
}
```

As we can see, the first five crabs are each having a different species, and all of them have been caught with Trap ID = 1. Now let's go back to the general insight page and fetch the data one more time.

- After:

This time, the data is fetched successfully

<div><div></div><div>TRAP_LOCATION</div><div>PHONE</div></div>				
Fetch Data				
TRAPID	BAITTYPE	TRAPTYPE	TRAP_LOCATION	PHONE
14	Chicken	Clamshell	East	374-939-2754
13	Chicken	Clamshell	North	374-939-2754
15	Chicken	Clamshell	South	185-274-5245
17	Chicken	Clamshell	East	245-924-5203
8	Chicken	Circle	South	984-921-5234
26	Chicken	Clamshell	East	984-921-5234
34	Chicken	Clamshell	Main	245-924-5203
50	Chicken	Clamshell	Main	316-274-3485
11	Chicken	Clamshell	Main	316-274-3485
IDs of traps that have caught all 5 types of crabs:				
Trap ID				
1				

And as we can see, after all the trap tuples, we have another section displaying all the trapIDs that caught all five kinds of traps. As we expected, only the trap with trapID = 1 is being displayed.