

CPSC 304 Project Cover Page

Milestone #: 4

Date: November 25, 2022

Group Number: 83

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Vlad Lungu	37265155	z1f2m	v_lungu@hotmail.com
Alice Zhao	60870714	u1i9m	azhao991@gmail.com
Kyle Schellen	95809380	d1u2w	kyle.schellen@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 e-mail 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/CPSC304-2022W-T1/project_d1u2w_u1i9m_z1f2m.git

Beaver Scout Management Program (BSMP)

This application stores information on Beaver Scouts, a co-ed youth organization that offers children indoor programming and outdoor adventures to prepare them for success in the world.

BSMP can be used by scout leaders and **has the potential** to

- track beaver, guardian, and club leader personal information
- track leader training, roles, and privileges within the colony
- add and remove badges, activities, and meetings
- monitor past activities and meetings and who was leading and in attendance
- maintain information on skits, songs, crafts, and meeting locations
- filter through and select events to easily retrieve information related to the event

Important Note: The team decided to focus on the Beaver and Guardian tables to generate the GUI for simplicity and time constraints which still provides similar functionality as listed above. An user will be able to:

- Create, Edit, and Update a new Beaver and Guardian
- View all data associated with a specific Beaver and Guardian
- Filter Beavers and guardians according to some specified parameter or attribute
- Input new Beaver and Guardian data

Updates to schema:

- Modified Activity's completionTime attribute to be completionDate to fulfil the GUI requirements in Milestone 3. ie. to be able to easily display the completion dates of activity in the Log table; however, this was not needed for our GUI as we decided to pursue a different route.
- Modified our FD's as we decomposed incorrectly, which caused our decomposed schemas to be altered. The changes are reflected below.
 - TailColor no longer has a FK as Beaver has the FK referencing TailColor
 - Beaver now references TailColor's grade
 - MaxSupervised no longer has a FK for yearsAsLeader as Leader has the FK referencing MaxSupervised

- AccessToBankAccount no longer has a FK for role as Leader has the FK referencing AccessToBankAccount
- Leader now has FKs referencing MaxSupervised(yearsAsLeader) and AccessToBankAccount(role)
- Included domains in each schema to align with the suggestions in previous milestones
- As we were building our tables and designing our GUI, we noticed that certain attributes would be better represented with NOT NULL constraints because tables with NULL fields was not an optimal design in select cases. Ex. having a guardian with no name.

SCHEMA LEGEND

BOLD	Foreign Keys
<u>UNDERLINE</u>	Primary Keys
<i>ITALICS</i>	Candidate Keys
<i>BLUE TEXT</i>	Schema Modifications

The updated schema are as follows:

TailColor(grade: CHAR(2), tailColor: VARCHAR(20))

Beaver(name: VARCHAR(20), email: VARCHAR(50), **grade**: CHAR(2))

- Email REFERENCES Guardian(email)
- Grade REFERENCES TailColor(grade)

MaxSupervised(yearsAsLeader: INT, maxSupervised: INT)

AccessToBankAccount(role: VARCHAR(20), bankPrivilege: VARCHAR(20))

Leader(colonyName: VARCHAR(20), name: VARCHAR(50), trainingCompleted: BOOLEAN, firstAid: BOOLEAN, policeCheckCompleted: BOOLEAN, phoneNumber: CHAR(10), email: VARCHAR(50), **yearsAsLeader**: INT, **role**: VARCHAR(20))

- yearsAsLeader REFERENCES MaxSupervised(yearsAsLeader)
- role REFERENCES AccessToBankAccount(role)

Activity(title: VARCHAR(50), completionDate: DATE)

- Constraint: partial disjoint on Songs/Skits

Guardian(email: VARCHAR(50), name VARCHAR(50), *phoneNumber*: CHAR(10))

- *phoneNumber* is UNIQUE
- Name NOT NULL

All remaining schemas (below) remain the same:

Badge(name: VARCHAR(50), requirements: VARCHAR)

Songs(title: VARCHAR(50), lyrics: TEXT, type: VARCHAR(50))

- Title REFERENCES Activity(title)

Skits(title: VARCHAR, actingRoles: VARCHAR, script: VARCHAR)

- Title REFERENCES Activity(title)

Location(address: VARCHAR(100), name: VARCHAR(50), outdoors: BOOLEAN)

Craft(name: VARCHAR(50), instructions: TEXT, completionTime: INT)

Learn(name: VARCHAR(20), email: VARCHAR(50), title: VARCHAR(50))

- Name, email REFERENCES Beaver(name, email)
- Title references Activity(title)

BeaverBadgeProgress(beaverName: VARCHAR(20), email: VARCHAR(50),
badgeName: VARCHAR(50), dateCompleted: DATE)

- beaverName, email REFERENCES Beaver(name, email)
- badgeName REFERENCES Badge(name)

Meeting(address: VARCHAR(100), name: VARCHAR(50), date: DATE,
leadersAvailable: INT)

- as this is a many to one relationship, we are combining Meeting and TakePlaceAt into Meeting
- address, name REFERENCES Location(address, name)
- participation constraint: address and locationName cannot be null
- participation constraint: Given that we have a many to many total participation of meeting with both leaders and beavers, we acknowledge:
 - In the Organize table, there must be at least 1 leader for a meeting date. If there are no leaders, meeting will be deleted.
 - In the Attend table, there must be at least 1 beaver for a meeting date. If there are no beavers, meeting will be deleted.

MeetingAttendance(name: VARCHAR(50), email: VARCHAR(50), date: DATE, duesPaid: BOOLEAN)

- Email REFERENCES Guardian(email)
- Date REFERENCES Meeting(date)

MeetingOrganization(date: DATE, colonyName: VARCHAR(20), responsibility: VARCHAR(50))

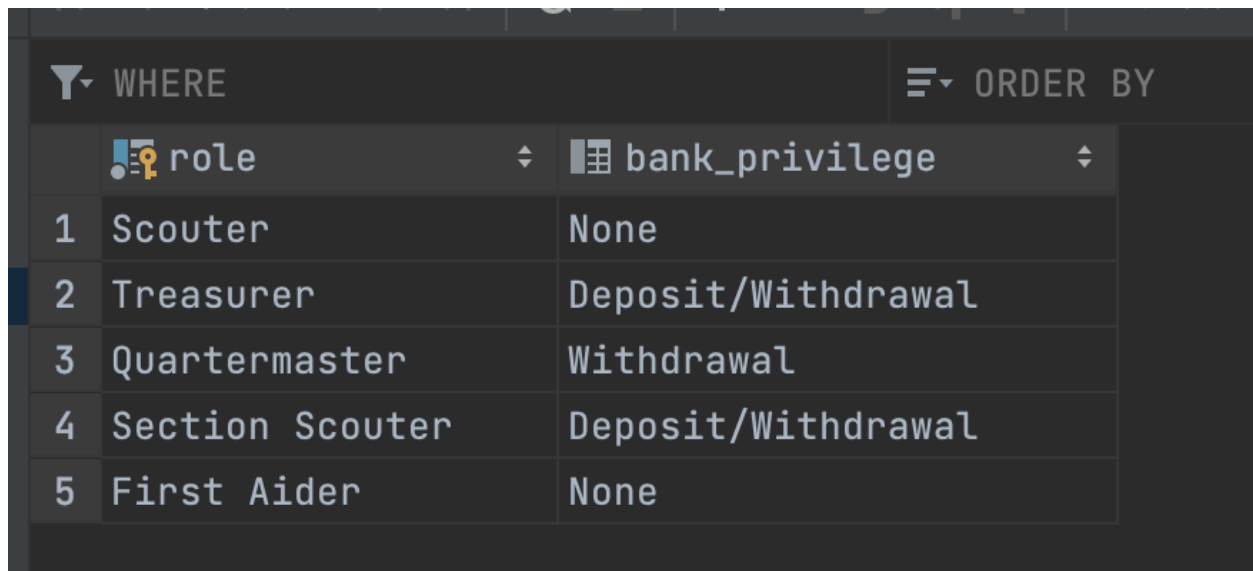
- Date REFERENCES Meeting(date)
- colonyName REFERENCES Leader(colonyName)

CraftAtMeeting(date: DATE, name: VARCHAR(50), cost: INT)

- date REFERENCES Meeting(date)
- name REFERENCES Craft(name)

SQL Data Screenshots:

AccessToBankAccount



The screenshot shows a SQL query result with two columns: 'role' and 'bank_privilege'. The results are ordered by role. The roles and their corresponding privileges are:

	role	bank_privilege
1	Scouter	None
2	Treasurer	Deposit/Withdrawal
3	Quartermaster	Withdrawal
4	Section Scouter	Deposit/Withdrawal
5	First Aider	None




Activity

WHERE		ORDER BY	
	🔑 title		📅 completion_date
1	Whistle		2020-05-06
2	Tag		2020-05-06
3	Hop Scotch		2020-06-06
4	Treasure Hunt		2021-07-08
5	Hide and Seek		2021-07-08
6	Firemaking		2021-09-09
7	Public Speaking		2022-08-09
8	Story Time		2021-04-06
9	Bird Watching		2022-10-10
10	Reading		2022-10-10
11	Jellyfish Song		2022-10-03
12	Bear Hunt Song		2022-10-01
13	Hokey Pokey		2022-10-10
14	Penguin song		2022-10-11
15	Vespers		2022-10-10
16	Invisible Bench		<null>
17	Submarine Skit		2022-10-10
18	Ugliest Man in the World		<null>
19	Bologna		<null>
20	Gotta Go Wee		2022-10-10

Badge

WHERE		ORDER BY
	name	requirements
1	Exploring Beaver	Go on 5 hikes
2	Musical Beaver	Perform a musical piece on an instrument
3	Aquatic Skills 1	Demonstrate appropriate water safety
4	Paddling Skills 1	Show knowledge of how to paddle in a canoe
5	Olympic Beaver	Participate in a sports team

Beaver

WHERE		ORDER BY	
	 name	 email	 grade
1	Jake	mr_jake@gmail.com	JK
2	Stacy	stacysMom@hotmail.com	02
3	Zuul	zuul@zuul.com	01
4	Jimothy	generic1997@gmail.com	SK
5	Alice	zuul@zuul.com	03
6	Kyle	zuul@zuul.com	01
7	Vlad	zuul@zuul.com	02
8	Chad	mr_jake@gmail.com	01
9	Chris	mr_jake@gmail.com	JK
10	Heather	mr_jake@gmail.com	02
11	Logan	stacysMom@hotmail.com	JK

BeaverBadgeProgress

WHERE		ORDER BY		
	beaver_name	email	badge_name	date_completed
1	Jake	mr_jake@gmail.com	Exploring Beaver	<null>
2	Stacy	stacysMom@hotmail.com	Musical Beaver	2022-09-12
3	Zuul	zuul@zuul.com	Aquatic Skills 1	2021-08-15
4	Jimothy	generic1997@gmail.com	Paddling Skills 1	<null>
5	Alice	zuul@zuul.com	Olympic Beaver	2022-11-01
6	Alice	zuul@zuul.com	Exploring Beaver	2021-01-01
7	Alice	zuul@zuul.com	Aquatic Skills 1	2021-07-01
8	Alice	zuul@zuul.com	Paddling Skills 1	2022-05-01
9	Alice	zuul@zuul.com	Musical Beaver	2020-10-01

Craft

WHERE		ORDER BY	
	name	instructions	completion_time
1	Paper Airplane	Get paper, fold plane	15
2	Compass	Put water in bottlecap, put needle in water	10
3	Gumball Caterpillar	Wrap gumballs in cellophane, add eyes	10
4	Tube Owl	Fold ears, Draw face on owl	15
5	Origami	Fold a crane	20

CraftAtMeeting

WHERE		ORDER BY	
	date	name	cost
1	2022-09-29	Paper Airplane	15
2	2022-10-06	Compass	30
3	2022-10-13	Gumball Caterpillar	25
4	2022-10-27	Tube Owl	50
5	2022-11-09	Origami	20

Guardian

WHERE		ORDER BY	
	email	name	phone_number
1	mr_jake@gmail.com	Dan Smith	4165552012
2	stacysMom@hotmail.com	Sam White	9055551121
3	zuul@zuul.com	Vinz Clortho	6665551010
4	generic1997@gmail.com	Jacob Anthony	6041231234
5	momomom@yahoo.ca	James James	1112223333

Leader

WHERE		ORDER BY							
	colony_name	name	training_completed	first_aid	police_check_completed	phone_number	email	years_as_leader	role
1	Rusty	Don Smith	• true	• true	• true	6045552323	don_smith@gmail.com	5	Treasurer
2	Acorn	Cathy Liu	• true	• false	• true	6045552223	cathy_liu@gmail.com	3	Scouter
3	Hopper	Don Smith	• false	• false	• true	6045552326	don_smith2@gmail.com	1	Scouter
4	Rainbow	Vlad Vlad	• true	• true	• false	6045551111	vlad@gmail.com	3	Scouter
5	Hawkeye	Derek Britton	• true	• true	• false	6045552723	hawkeye123@gmail.com	5	Quartermaster

Learn

WHERE		ORDER BY	
	name	email	title
1	Jake	mr_jake@gmail.com	Jellyfish Song
2	Stacy	stacysMom@hotmail.com	Invisible Bench
3	Zuul	zuul@zuul.com	Invisible Bench
4	Zuul	zuul@zuul.com	Submarine Skit
5	Jimothy	generic1997@gmail.com	Vespers

Location

WHERE		ORDER BY	
	address	name	outdoors
1	123 W 16th Avenue	Community Center	false
2	212 W 1st Avenue	Jericho Beach	• true
3	1455 Quebec Street	Science World	false
4	Imperial Drive	Camosun Bog	• true
5	3461 Ross Drive	UBC Farm	• true

MaxSupervised

WHERE		ORDER BY	
	👤 years_as_leader ↕	📊 max_supervised ↕	
1	0	5	
2	1	6	
3	2	8	
4	3	10	
5	5	10	

Meeting

WHERE		ORDER BY		
	📍 address ↕	👤 name ↕	📅 date ↕	📊 leaders_available ↕
1	123 W 16th Avenue	Community Center	2022-09-29	3
2	123 W 16th Avenue	Community Center	2022-10-06	2
3	123 W 16th Avenue	Community Center	2022-10-13	5
4	1455 Quebec Street	Science World	2022-10-27	2
5	212 W 1st Avenue	Jericho Beach	2022-11-09	4

MeetingAttendance

WHERE		ORDER BY		
	👤 name ↕	📧 email ↕	📅 date ↕	📊 dues_paid ↕
1	Jake	mr_jake@gmail.com	2022-09-29	• true
2	Jake	mr_jake@gmail.com	2022-10-06	false
3	Jake	mr_jake@gmail.com	2022-10-13	• true
4	Stacy	stacysMom@hotmail.com	2022-09-29	• true
5	Zuul	zuul@zuul.com	2022-09-29	• true

MeetingOrganization

WHERE			ORDER BY
	date	colony_name	responsibility
1	2022-09-29	Rusty	Craft supplies
2	2022-09-29	Rainbow	Teach song
3	2022-09-29	Hawkeye	Run opening and closing
4	2022-10-27	Acorn	Craft supplies
5	2022-11-09	Rainbow	Organize game

Skits

WHERE			ORDER BY
	title	acting_roles	script
1	Invisible Bench	Hey are you sitting on the invisible bench? Yes, come and j...	Beaver 1, Beaver 2, Beaver 3, Final Beaver
2	Submarine Skit	Once there was a submarine at sea...	Narrator, Line of Beavers, Final Beaver
3	Ugliest Man in the World	Come one, come all, to see the ugliest man in the world...	Ugliest man, Presenter, Beaver 1, Beaver 2, Scouter 1
4	Balogna	There are some very special echoes in these woods...	Scouter, Echo 1, Echo 2, Echo 3
5	Gotta Go Wee	Leader, I have to go wee...	Scouter, Beaver 1, Beaver 2, Beaver 3

Songs

WHERE			ORDER BY
	title	lyrics	type
1	Jellyfish Song	Arms up, wrist together Jellyfish Jellyfish	action and repeat after me
2	Bear Hunt Song	Going on a bear hunt, gonna catch a big one	action and repeat after me
3	Hokey Pokey	You put your right foot in, you put you right foot out...	action
4	Penguin song	Penguin attention, penguin salute	action and repeat after me
5	Vespers	Softly falls the light of day, as the campfire fades away	regular

TailColor

WHERE			ORDER BY
	grade	tail_color	
1	JK	No Tail	
2	SK	Brown	
3	01	Blue	
4	02	White	
5	03	Shooting Star	

Screenshots of every query

Query: INSERT: Code Implementation

Reference to INSERT in code: See path **packages/api/queries.js** Line: 78 -115

Note: Since beaver is a weak entity of guardian, before creating a new beaver, we first check if a guardian already exists with the same email. Else, we create a new guardian and new beaver.

INSERT INTO guardian(email, name, phone_number) values (\$1, \$2, \$3);

INSERT INTO beaver (name, email, grade) values (\$1, \$2, \$3);

```
const createGuardianSQL =
  "INSERT INTO guardian(email, name, phone_number) values ($1, $2, $3);";
const createBeaverSQL =
  "INSERT INTO beaver (name, email, grade) values ($1, $2, $3);";

const createBeaver = async (request, response) => {
  const { name, email, grade, phone, guardianName } = request.body;

  let existingGuardian;
  try {
    existingGuardian = await client.query(
      "SELECT * FROM guardian WHERE email = $1",
      [email]
    );
  } catch (error) {
    console.log(
      `Failed to find guardian for ${email}, attempting to create one...`
    );
  }

  if (!existingGuardian?.rows?.length) {
    client.query(
      createGuardianSQL,
      [email, guardianName, phone],
      (error, results) => {
        if (error) {
          throw error;
        }
      }
    );
  }
}
```

```

client.query(createBeaverSQL, [name, email, grade], (error, results) => {
  if (error) {
    throw error;
  }
});
};

```

Query: INSERT - Table **before** adding new Beaver/ new Guardian

Beaver table

Beavers

<input type="checkbox"/> Name	Email	Grade ↑
<input type="checkbox"/> Zuul	zuul@zuul.com	01
<input type="checkbox"/> Kyle	zuul@zuul.com	01
<input type="checkbox"/> Chad	mr_jake@gmail.com	01
<input type="checkbox"/> Stacy	stacysMom@hotmail.com	02
<input type="checkbox"/> Vlad	zuul@zuul.com	02
<input type="checkbox"/> Heather	mr_jake@gmail.com	02
<input type="checkbox"/> Alice	zuul@zuul.com	03
<input type="checkbox"/> Jake	mr_jake@gmail.com	JK
<input type="checkbox"/> Chris	mr_jake@gmail.com	JK
<input type="checkbox"/> Logan	stacysMom@hotmail.com	JK
<input type="checkbox"/> Jimothy	generic1997@gmail.com	SK

Rows per page: 25 1-11 of 11 < >

Guardian table

Guardians

<input type="checkbox"/> Email	Name	Phone_number
<input type="checkbox"/> mr_jake@gmail.com	Dan Smith	4165552012
<input type="checkbox"/> stacysMom@hotmail.com	Sam White	9055551121
<input type="checkbox"/> zuul@zuul.com	Vinz Clortho	6665551010
<input type="checkbox"/> generic1997@gmail.com	Jacob Anthony	6041231234
<input type="checkbox"/> momomom@yahoo.ca	James James	1112223333

Query: INSERT - Form to enter information for new Beaver (Carol Smith) and her Guardian (Kyle Smith)

GET GUARDIANS OF

GET GRADE STATISTICS

Email	Grade
zuul@zuul.com	01
zuul@zuul.com	01
mr_jake@gmail.co	01
stacysMom@hotmail.com	02
zuul@zuul.com	02
mr_jake@gmail.co	02
zuul@zuul.com	03
mr_jake@gmail.co	JK

Create New Beaver

Fill in beaver information

Name *

Carol Smith

Grade

01

Fill in guardian information

Name

Kyle Smith

Phone

7781234126

Email *

abcd@gmail.com

ADD >

Query: INSERT - table **after** adding new Beaver/ new Guardian

current: 12 rows.

Added beaver in row 4

Beavers			
<input type="checkbox"/> Name	Email	Grade ↑	
<input type="checkbox"/> Zuul	zuul@zuul.com	01	
<input type="checkbox"/> Kyle	zuul@zuul.com	01	
<input type="checkbox"/> Chad	mr_jake@gmail.com	01	
<input type="checkbox"/> Carol Smith	abcd@gmail.com	01	
<input type="checkbox"/> Stacy	stacysMom@hotmail.com	02	
<input type="checkbox"/> Vlad	zuul@zuul.com	02	
<input type="checkbox"/> Heather	mr_jake@gmail.com	02	
<input type="checkbox"/> Alice	zuul@zuul.com	03	
<input type="checkbox"/> Jake	mr_jake@gmail.com	JK	
<input type="checkbox"/> Chris	mr_jake@gmail.com	JK	
<input type="checkbox"/> Logan	stacysMom@hotmail.com	JK	
<input type="checkbox"/> Jimothy	generic1997@gmail.com	SK	
			Rows per page: 25 ▾ 1-12 of 12 < >

Added guardian in row 6

Guardians		
<input type="checkbox"/> Email	Name	Phone_number
<input type="checkbox"/> mr_jake@gmail.com	Dan Smith	4165552012
<input type="checkbox"/> stacysMom@hotmail.com	Sam White	9055551121
<input type="checkbox"/> zuul@zuul.com	Vinz Clortho	6665551010
<input type="checkbox"/> generic1997@gmail.com	Jacob Anthony	6041231234
<input type="checkbox"/> momomom@yahoo.ca	James James	1112223333
<input type="checkbox"/> abcd@gmail.com	Kyle Smith	7781234126

Query: DELETE: Code Implementation

Reference to DELETE in code: See path **packages/api/queries.js** Line: 66 -76

Note: Deleting a guardian will delete all beavers. (cascade on delete in schema)

DELETE FROM guardian WHERE email = \$1

```
const deleteGuardian = (request, response) => {
  client.query(
    "DELETE FROM guardian WHERE email = $1",
    [request.params.email],
    (error, results) => {
      if (error) {
        throw error;
      }
    }
  );
};
```

Query: DELETE - table before deleting Guardian (Kyle Smith + Carol Smith)

Beavers

<input type="checkbox"/> Name	Email	Grade ↑
<input type="checkbox"/> Zuul	zuul@zuul.com	01
<input type="checkbox"/> Kyle	zuul@zuul.com	01
<input type="checkbox"/> Chad	mr_jake@gmail.com	01
<input type="checkbox"/> Stacy	stacysMom@hotmail.com	02
<input type="checkbox"/> Vlad	zuul@zuul.com	02
<input type="checkbox"/> Heather	mr_jake@gmail.com	02
<input type="checkbox"/> Alice	zuul@zuul.com	03
<input type="checkbox"/> Jake	mr_jake@gmail.com	JK
<input type="checkbox"/> Chris	mr_jake@gmail.com	JK
<input type="checkbox"/> Logan	stacysMom@hotmail.com	JK
<input type="checkbox"/> Jimothy	generic1997@gmail.com	SK

Rows per page: 25 1-11 of 11 < >

Guardians

<input type="checkbox"/> Email	Name	Phone_number
<input type="checkbox"/> mr_jake@gmail.com	Dan Smith	4165552012
<input type="checkbox"/> stacysMom@hotmail.com	Sam White	9055551121
<input type="checkbox"/> zuul@zuul.com	Vinz Clortho	6665551010
<input type="checkbox"/> generic1997@gmail.com	Jacob Anthony	6041231234
<input type="checkbox"/> momomom@yahoo.ca	James James	1112223333
<input type="checkbox"/> abcd@gmail.com	Kyle Smith	7781234126

Query: DELETE - Guardian (Kyle Smith)

1 selected			 
<input type="checkbox"/> Email	Name	Phone_number	Delete
<input type="checkbox"/> mr_jake@gmail.com	Dan Smith	4165552012	
<input type="checkbox"/> stacysMom@hotmail.com	Sam White	9055551121	
<input type="checkbox"/> zuul@zuul.com	Vinz Clortho	6665551010	
<input type="checkbox"/> generic1997@gmail.com	Jacob Anthony	6041231234	
<input type="checkbox"/> momomom@yahoo.ca	James James	1112223333	
<input checked="" type="checkbox"/> abcd@gmail.com	Kyle Smith	7781234126	

Query: DELETE - table after deleting Guardian (Kyle Smith, abcd@gmail.com)

Note: deleting guardian (Kyle Smith) also deletes Beaver (Carol Smith)

Guardians			
<input type="checkbox"/> Email	Name	Phone_number	
<input type="checkbox"/> mr_jake@gmail.com	Dan Smith	4165552012	
<input type="checkbox"/> stacysMom@hotmail.com	Sam White	9055551121	
<input type="checkbox"/> zuul@zuul.com	Vinz Clortho	6665551010	
<input type="checkbox"/> generic1997@gmail.com	Jacob Anthony	6041231234	
<input type="checkbox"/> momomom@yahoo.ca	James James	1112223333	
Rows per page: 5 ▾ 1-5 of 5 < >			

<input type="checkbox"/> Name	Email	Grade ↑
<input type="checkbox"/> Zuul	zuul@zuul.com	01
<input type="checkbox"/> Kyle	zuul@zuul.com	01
<input type="checkbox"/> Chad	mr_jake@gmail.com	01
<input type="checkbox"/> Stacy	stacysMom@hotmail.com	02
<input type="checkbox"/> Vlad	zuul@zuul.com	02
<input type="checkbox"/> Heather	mr_jake@gmail.com	02
<input type="checkbox"/> Alice	zuul@zuul.com	03
<input type="checkbox"/> Jake	mr_jake@gmail.com	JK
<input type="checkbox"/> Chris	mr_jake@gmail.com	JK
<input type="checkbox"/> Logan	stacysMom@hotmail.com	JK
<input type="checkbox"/> Jimothy	generic1997@gmail.com	SK

Rows per page: 25 1-11 of 11 < >

Query: UPDATE: Code Implementation

Reference to UPDATE in code: See path `packages/api/queries.js` Line: 117 -138

UPDATE beaver SET grade = \$1, name = \$2 WHERE name = \$3 AND email = \$4

```
const editBeaver = (request, response) => {
  const id = request.params.id;
  const [nameId, emailId] = id.split("-");
  const { grade, name } = request.body;

  client.query(
    "UPDATE beaver SET grade = $1, name = $2 WHERE name = $3 AND email = $4",
    [grade, name, nameId, emailId]
  );
};
```

Query: UPDATE: Before table with Beaver Zuul with grade 1

<input type="checkbox"/> Name	Email	Grade ↑
<input type="checkbox"/> Zuul	zuul@zuul.com	01

Query: UPDATE: Form to update Beaver Zuul to grade JK

Edit Beaver

Fill in beaver information

Name*

Grade

EDIT ➤

Name	Email	Grade
Zuul	zuul@zuul.com	JK
Jimothy	generic1997@gmail.com	SK

Query: UPDATE: After table when Zuul is updated to JK

<input type="checkbox"/>	Zuul	zuul@zuul.com	JK
<input type="checkbox"/>	Jimothy	generic1997@gmail.com	SK

Rows per page: 25 1-11 of 11

Query: SELECTION: Code Implementation

Reference to SELECTION in code: See path `packages/api/queries.js` Line: 15 - 38

SELECT \${columns} FROM beaver \${whereClause}

Note: `columns` is variable that is a comma separated string representing the selected table column headers

`whereClause` is a variable that represents WHERE in sql to input certain parameters to select on

```
const getBeavers = (request, response) => {
  const columns = getQueryParam(request, "columns") ?? "*";

  const column = getQueryParam(request, "column");
  const rawOperator = getQueryParam(request, "operator");
  const value = getQueryParam(request, "value");
  let whereClause = "";
  if (column && rawOperator && value) {
    const operator = OPERATORS[rawOperator];
    if (operator) {
      whereClause = `WHERE ${column} ${operator} '${value}'`;
    }
  }

  client.query(
    `SELECT ${columns} FROM beaver ${whereClause}`,
    (error, results) => {
      if (error) {
        throw error;
      }
      response.status(200).json(results.rows);
    }
  );
};
```

Query: SELECTION: Form to select any column and any value from Beaver table. (i.e select all rows with email = zuul@zuul.com)

The screenshot shows a web application interface. On the left, a vertical list of email addresses is visible: zuul@zuul.com, mr_jake@gmail.com, zuul@zuul.com, mr_jake@gmail.com, mr_jake@gmail.com, stacysMom@hotmail.com, and zuul@zuul.com. A modal dialog box titled "Filter Columns" is centered on the screen. It contains three input fields: "Column" with a dropdown menu showing "email", "Operator" with a dropdown menu showing "equals", and "Value" with a text input field containing "zuul@zuul.com". Below these fields is a red button labeled "FILTER" with a small icon. The background of the application is a light gray.

Query: SELECTION: After table with only rows with email = zuul@zuul.com

Beavers			
<input type="checkbox"/> Name	Email	Grade ↑	
<input type="checkbox"/> Kyle	zuul@zuul.com	01	
<input type="checkbox"/> Vlad	zuul@zuul.com	02	
<input type="checkbox"/> Alice	zuul@zuul.com	03	
<input type="checkbox"/> Zuul	zuul@zuul.com	JK	
			Rows per page: 25 ▾ 1-4 of 4 < >

Query: PROJECTION: Code Implementation

Reference to PROJECTION in code: See path **packages/api/queries.js** Line: 15 - 38
See "columns" on line 16 and 30

SELECT \${columns} FROM beaver \${whereClause}

Note: `columns` is variable that is a comma separated string representing the selected table column headers

`whereClause` is a variable that represents WHERE in sql to input certain parameters to select on

```
const getBeavers = (request, response) => {
  const columns = getQueryParam(request, "columns") ?? "*";

  const column = getQueryParam(request, "column");
  const rawOperator = getQueryParam(request, "operator");
  const value = getQueryParam(request, "value");
  let whereClause = "";

  if (column && rawOperator && value) {
    const operator = OPERATORS[rawOperator];
    if (operator) {
      whereClause = `WHERE ${column} ${operator} '${value}'`;
    }
  }

  client.query(
    `SELECT ${columns} FROM beaver ${whereClause}`,
    (error, results) => {
      if (error) {
        throw error;
      }

      response.status(200).json(results.rows);
    }
  );
};
```

Query: PROJECTION: Form to select which columns to project on

Beavers			
<input type="checkbox"/> Name	Email	Grade	
<input type="checkbox"/> Kyle	zuul@zuul.com	01	
<input type="checkbox"/> Chad	mr_jake@gmail.co	01	
<input type="checkbox"/> Stacy	stacysMom@hotmail	02	
<input type="checkbox"/> Vlad	zuul@zuul.com	02	

Select Columns

☒ name

☒ email

☒ grade

Query: PROJECTION: After table, select only email and grade

Beavers	
<input type="checkbox"/> Email	Grade ↑
<input type="checkbox"/> zuul@zuul.com	01
<input type="checkbox"/> mr_jake@gmail.com	01
<input type="checkbox"/> stacysMom@hotmail.com	02
<input type="checkbox"/> zuul@zuul.com	02
<input type="checkbox"/> mr_jake@gmail.com	02
<input type="checkbox"/> zuul@zuul.com	03
<input type="checkbox"/> mr_jake@gmail.com	JK
<input type="checkbox"/> mr_jake@gmail.com	JK
<input type="checkbox"/> stacysMom@hotmail.com	JK
<input type="checkbox"/> zuul@zuul.com	JK
<input type="checkbox"/> generic1997@gmail.com	SK
Rows per page: 25 ▾ 1–11 of 11 < >	

Query: JOIN: Code Implementation

Reference to JOIN in code: See path **packages/api/queries.js** Line: 40 - 54

Joins beaver with guardian and with tail color to get Beaver information

SELECT guardian.name as guardianName, phone_number, tail_color FROM beaver FULL JOIN guardian ON beaver.email = guardian.email FULL JOIN tailcolor ON beaver.grade = tailcolor.grade WHERE beaver.name = \$1 AND beaver.email = \$2

```
const getBeaver = (request, response) => {
  const id = request.params.id;
  const [name, email] = id.split("-");

  client.query(
    `SELECT guardian.name as guardianName, phone_number, tail_color FROM beaver FULL
JOIN guardian ON beaver.email = guardian.email FULL JOIN tailcolor ON beaver.grade =
tailcolor.grade WHERE beaver.name = $1 AND beaver.email = $2 `,
    [name, email],
    (error, results) => {
      if (error) {
        throw error;
      }
    }
  );
}
```

```

    }
    response.status(200).json(results.rows);
  }
};
};

```

Query: JOIN: When any Beaver is selected, a join table shows with guardian and tail color info

1 selected			
<input type="checkbox"/>	Name	Email	Grade ↑
<input type="checkbox"/>	Kyle	zuul@zuul.com	01
<input type="checkbox"/>	Chad	mr_jake@gmail.com	01
<input type="checkbox"/>	Stacy	stacysMom@hotmail.com	02
<input type="checkbox"/>	Vlad	zuul@zuul.com	02
<input checked="" type="checkbox"/>	Heather	mr_jake@gmail.com	02

Guardian
Dan Smith (4165552012)
Tail Colour
White

Query: AGGREGATION GROUP BY: Code Implementation

Reference to AGGREGATION GROUP BY in code: See path **packages/api/queries.js** Line: 128-138

Beavers grouped by grade to get grade statistics of count of beavers for each grade.

SELECT grade, count(*) FROM beaver GROUP BY grade

```

const getGradeStatistics = (request, response) => {
  client.query(
    "SELECT grade, count(*) FROM beaver GROUP BY grade",
    (error, results) => {
      if (error) {
        throw error;
      }
      response.status(200).json(results.rows);
    }
  );
};

```


Query: AGGREGATION GROUP BY: Button: Get Grade Statistics displays statistics table

GET OVERWORKED GUARDIANS ↗	GET GUARDIANS OF EAGER BEAVERS ↗	GET COMPLETIONIST BEAVERS ↗	GET GRADE STATISTICS ↗	ADD BEAVER 👤
Grade Statistics				
grade	count			
SK	1			
03	1			
01	2			
02	3			
JK	4			

Query: AGGREGATION HAVING: Code Implementation

Reference to AGGREGATION HAVING in code: See path **packages/api/queries.js** Line: 140-150

Get aggregation of guardians with more than 1 beaver

SELECT guardian.name, count(beaver.name) as childCount FROM guardian FULL JOIN beaver ON guardian.email = beaver.email GROUP BY guardian.name HAVING count(beaver.name) > 1

```
const getOverworkedGuardians = (request, response) => {
  client.query(
    "SELECT guardian.name, count(beaver.name) as childCount FROM guardian FULL JOIN
beaver ON guardian.email = beaver.email GROUP BY guardian.name HAVING
count(beaver.name) > 1",
    (error, results) => {
      if (error) {
        throw error;
      }
      response.status(200).json(results.rows);
    }
  );
};
```

Query: AGGREGATION HAVING: Button: Get Overworked Guardians

3 guardians who has over 1 beavers in the system

GET OVERWORKED GUARDIANS ↗

GET GUARDIANS OF EAGER BEAVERS ↗

GET COMPLETIONIST BEAVERS ↗

Overworked Guardians

name	childcount
Sam White	2
Dan Smith	4
Vinz Clortho	4

Query: NESTED AGGREGATION: Code Implementation

Reference to NESTED AGGREGATION in code: See path **packages/api/queries.js** Line: 152-171

- Aggregation to get guardians with beavers who has completed at least 1 badge

SELECT guardian.name, count(badge.name) as badge_count FROM guardian JOIN badge ON badge.name IN (SELECT badge_name FROM beaverbadgeprogress WHERE beaverbadgeprogress.email = guardian.email AND date_completed IS NOT NULL) GROUP BY guardian.name

```
const getGuardiansWithEagerBeavers = (request, response) => {
  client.query(
    "SELECT guardian.name, count(badge.name) as badge_count FROM guardian JOIN badge ON
    badge.name IN (SELECT badge_name FROM beaverbadgeprogress WHERE
    beaverbadgeprogress.email = guardian.email AND date_completed IS NOT NULL) GROUP BY
    guardian.name",
    (error, results) => {
      if (error) {
        throw error;
      }
      response.status(200).json(results.rows);
    }
  );
};
```

Query: NESTED AGGREGATION: Button: Get Guardians with Eager Beavers

2 guardians with beavers who have completed at least 1 badge

GET OVERWORKED GUARDIANS ↕

GET GUARDIANS OF EAGER BEAVERS ↕

GET COMPLETIONIST BEAVERS ↕

Guardians With Eager Beavers

name	badge_count
Sam White	1
Vinz Clortho	5

Query: DIVISION: Code Implementation

Reference to Division in code: See path **packages/api/queries.js** Line: 173 -183

Getting beavers who has completed all badges

```
SELECT * FROM beaver WHERE NOT EXISTS (SELECT badge.name FROM badge  
WHERE NOT EXISTS (SELECT bbp.email FROM beaverbadgeprogress bbp WHERE  
bbp.badge_name = badge.name AND bbp.email = beaver.email AND bbp.beaver_name =  
beaver.name))
```

```
getCompletionistBeavers = (request, response) => {  
  client.query(  
    "SELECT * FROM beaver WHERE NOT EXISTS (SELECT badge.name FROM badge WHERE NOT  
EXISTS (SELECT bbp.email FROM beaverbadgeprogress bbp WHERE bbp.badge_name =  
badge.name AND bbp.email = beaver.email AND bbp.beaver_name = beaver.name))",  
    (error, results) => {  
      if (error) {  
        throw error;  
      }  
      response.status(200).json(results.rows);  
    }  
  );  
};
```

Query: DIVISION: Button: Get Completionist Beavers

1 beaver Alice who has completed all badges

GET OVERWORKED GUARDIANS ↕

GET GUARDIANS OF EAGER BEAVERS ↕

GET COMPLETIONIST BEAVERS ↕

GET GRADE STATISTICS ↕

ADD BEAVER ➕

Completionist Beavers

name	email	grade
Alice	zuul@zuul.com	03