University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 4

Date: Nov 28, 2023

Group Number: 55

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Helen Luo	28494193	l2s4b	hluo301@gmail.com
Timothy Jin	14651228	w3y3p	timothy.jin26@gmail.com
Nic Ung	67524991	q5y7r	nicholasung.edu@gmail.com

University of British Columbia, Vancouver
Department of Computer Science

1. Summary

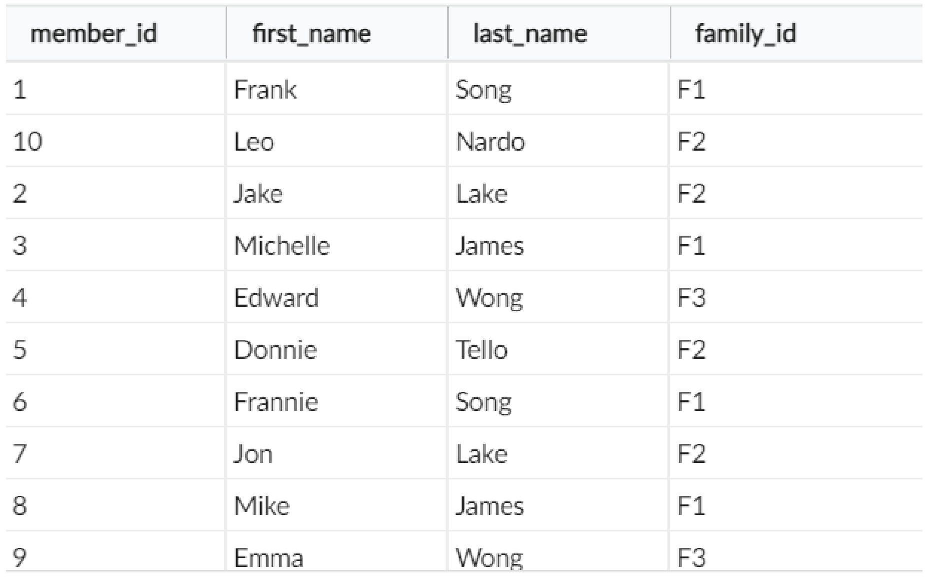
Oftentimes, parents come to pick up their children at the childcare center and ask what they have done for the day. The database models what the child does in the childcare, such as what activity they did or what food they had for lunch. It also helps the childcare keep track of the families that it services and the caretakers that it hires.

2. Schema Changes

None.

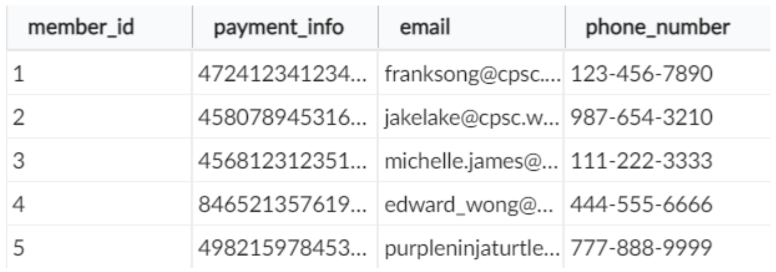
3. Schema and Screenshots

family_member(first_name: VARCHAR(40), last_name: VARCHAR(40), member_id: VARCHAR(20), **family_id: VARCHAR(20)**)



member_id	first_name	last_name	family_id
1	Frank	Song	F1
10	Leo	Nardo	F2
2	Jake	Lake	F2
3	Michelle	James	F1
4	Edward	Wong	F3
5	Donnie	Tello	F2
6	Frannie	Song	F1
7	Jon	Lake	F2
8	Mike	James	F1
9	Emma	Wong	F3

parent(**member_id: VARCHAR(20)**, payment_info: VARCHAR(20), email: VARCHAR(250), phone_number: VARCHAR(20))



member_id	payment_info	email	phone_number
1	472412341234...	franksong@cpsc....	123-456-7890
2	458078945316...	jakelake@cpsc.w...	987-654-3210
3	456812312351...	michelle.james@...	111-222-3333
4	846521357619...	edward_wong@...	444-555-6666
5	498215978453...	purpleninjaturtle...	777-888-9999

University of British Columbia, Vancouver
Department of Computer Science

child(**member_id: VARCHAR(20)**, **birthday: DATE**, **schedule_id: VARCHAR(20)**, **meal_name**: VARCHAR(20), meal_time: TIME)

member_id	birthday	schedule_id	meal_name	meal_time
10	2016-05-10	S2	Lunch	12:30:00
6	2015-03-12	S1	Lunch	12:00:00
7	2017-08-25	S2	Dinner	18:30:00
8	2014-11-05	S1	Breakfast	08:00:00
9	2019-02-20	S3	Snack	15:30:00

birthday_to_age(birthday: DATE, age: INT)

birthday	age
2014-11-05	8
2015-03-12	8
2016-05-10	7
2017-08-25	6
2019-02-20	4

family(family_id: VARCHAR(20), address: VARCHAR(250), **branch_id: VARCHAR(20)**, Fee: INT)

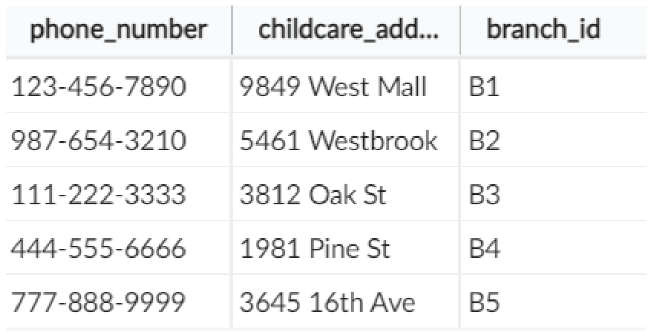
family_id	family_address	branch_id	fee
F1	1234 Road Place	B1	200
F2	4567 Marine Dri...	B2	400
F3	7894 Oak St	B1	400
F4	1560 Pine St	B3	550
F5	4895 41st Ave	B2	180

childcare_branch_room(room_capacity: INT, room_number: INT, **branch_id: VARCHAR(20)**)

room_number	room_capacity	branch_id
101	30	B1
102	25	B1
201	20	B2
202	15	B2
301	30	B3

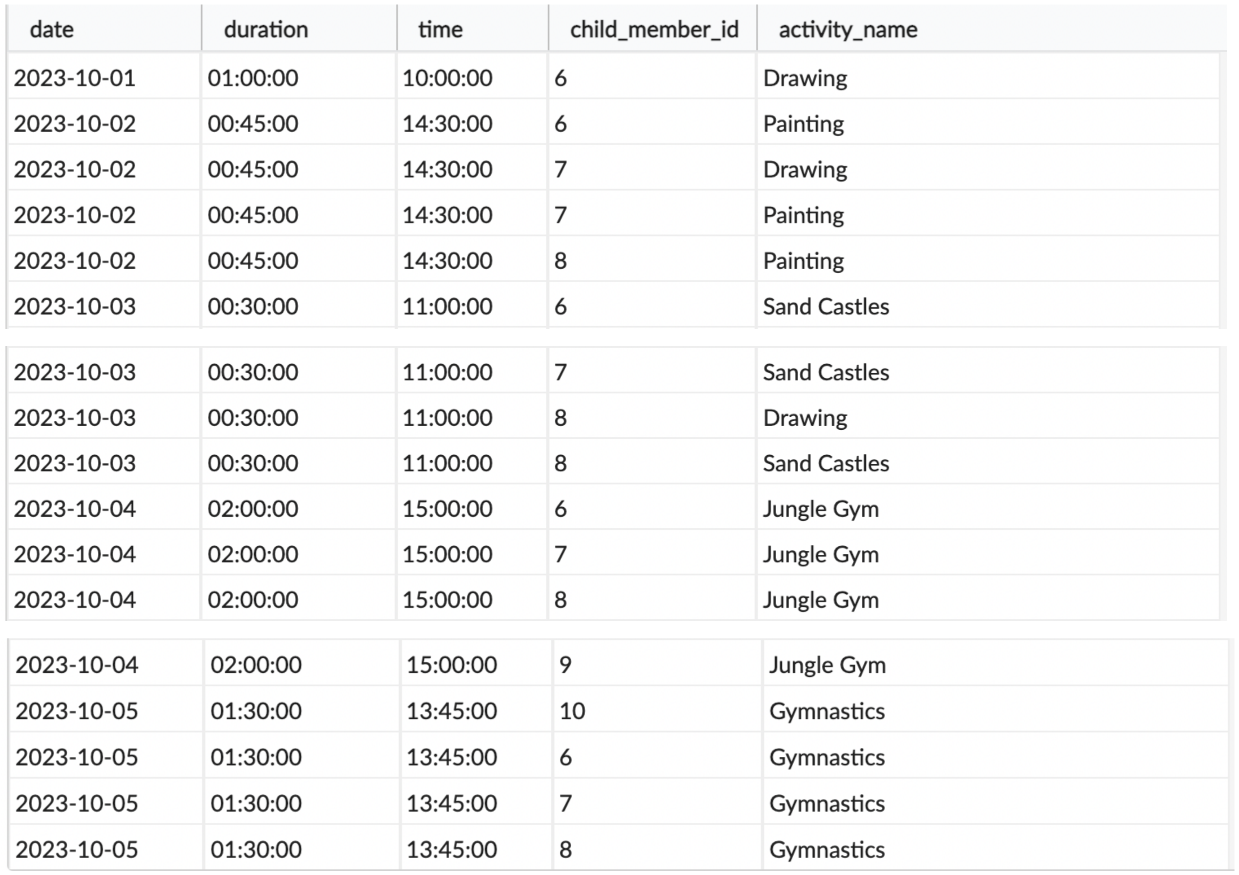
University of British Columbia, Vancouver
Department of Computer Science

childcare_branches(phone_number: VARCHAR(20), address: VARCHAR(250), branch_id: VARCHAR(20))



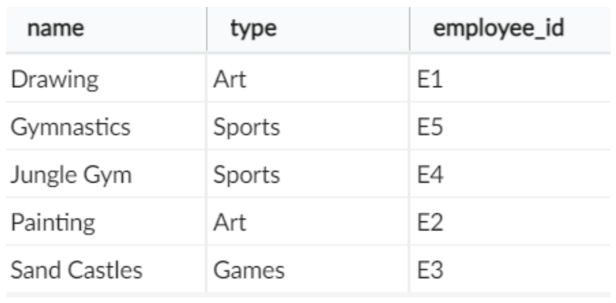
phone_number	childcare_add...	branch_id
123-456-7890	9849 West Mall	B1
987-654-3210	5461 Westbrook	B2
111-222-3333	3812 Oak St	B3
444-555-6666	1981 Pine St	B4
777-888-9999	3645 16th Ave	B5

child_does_activity(date: DATE, duration:TIME, time: TIME, **child_member_id: VARCHAR(20)**, **activity_name**: VARCHAR(250))



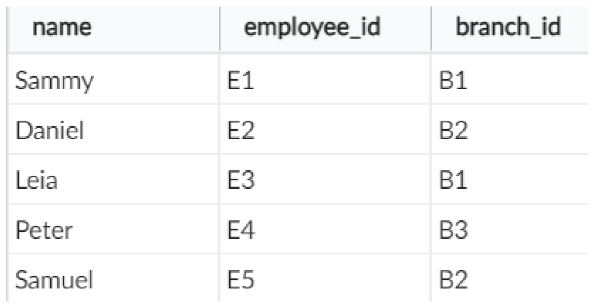
date	duration	time	child_member_id	activity_name
2023-10-01	01:00:00	10:00:00	6	Drawing
2023-10-02	00:45:00	14:30:00	6	Painting
2023-10-02	00:45:00	14:30:00	7	Drawing
2023-10-02	00:45:00	14:30:00	7	Painting
2023-10-02	00:45:00	14:30:00	8	Painting
2023-10-03	00:30:00	11:00:00	6	Sand Castles
2023-10-03	00:30:00	11:00:00	7	Sand Castles
2023-10-03	00:30:00	11:00:00	8	Drawing
2023-10-03	00:30:00	11:00:00	8	Sand Castles
2023-10-04	02:00:00	15:00:00	6	Jungle Gym
2023-10-04	02:00:00	15:00:00	7	Jungle Gym
2023-10-04	02:00:00	15:00:00	8	Jungle Gym
2023-10-04	02:00:00	15:00:00	9	Jungle Gym
2023-10-05	01:30:00	13:45:00	10	Gymnastics
2023-10-05	01:30:00	13:45:00	6	Gymnastics
2023-10-05	01:30:00	13:45:00	7	Gymnastics
2023-10-05	01:30:00	13:45:00	8	Gymnastics

activity(name: VARCHAR(250), type: VARCHAR(250), **employee_id: VARCHAR(20)**)



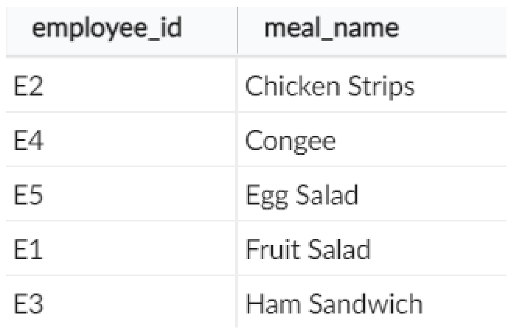
name	type	employee_id
Drawing	Art	E1
Gymnastics	Sports	E5
Jungle Gym	Sports	E4
Painting	Art	E2
Sand Castles	Games	E3

caretaker(name: VARCHAR(40), employee_id: VARCHAR(20), **branch_id: VARCHAR(20)**)



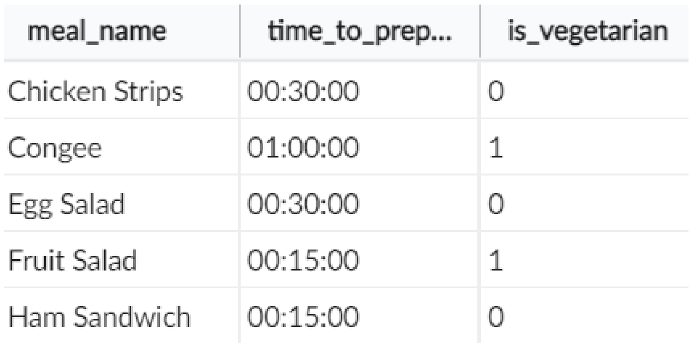
name	employee_id	branch_id
Sammy	E1	B1
Daniel	E2	B2
Leia	E3	B1
Peter	E4	B3
Samuel	E5	B2

caretaker_prepare_lunch(employee_id: VARCHAR(20), meal_name: VARCHAR(20))



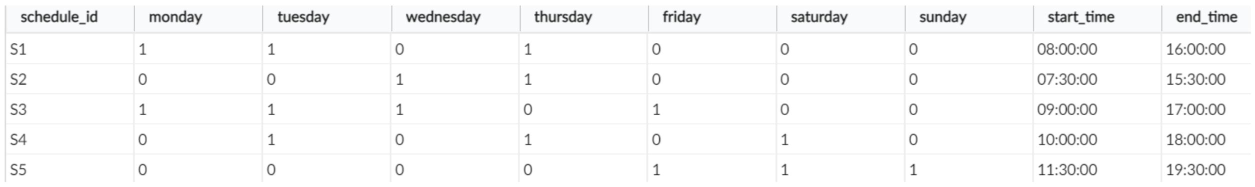
employee_id	meal_name
E2	Chicken Strips
E4	Congee
E5	Egg Salad
E1	Fruit Salad
E3	Ham Sandwich

lunch_option(meal_name: VARCHAR(100), time_to_prepare: TIME, is_vegetarian: BOOL)



meal_name	time_to_prep...	is_vegetarian
Chicken Strips	00:30:00	0
Congee	01:00:00	1
Egg Salad	00:30:00	0
Fruit Salad	00:15:00	1
Ham Sandwich	00:15:00	0

schedule(schedule_id: VARCHAR(20), monday: BOOL, tuesday: BOOL, wednesday: BOOL, thursday: BOOL, friday: BOOL, saturday: BOOL, sunday: BOOL, start_time: TIME, end_time: TIME)



schedule_id	monday	tuesday	wednesday	thursday	friday	saturday	sunday	start_time	end_time
S1	1	1	0	1	0	0	0	08:00:00	16:00:00
S2	0	0	1	1	0	0	0	07:30:00	15:30:00
S3	1	1	1	0	1	0	0	09:00:00	17:00:00
S4	0	1	0	1	0	1	0	10:00:00	18:00:00
S5	0	0	0	0	1	1	1	11:30:00	19:30:00

4. All SQL Queries

All SQL Queries are found on the Amazon AWS website, in the lambda function.

INSERT

# Add an activity based on available employee ids and user input
Frontend Code:
github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/childcare-website/src/Components/InsertActivity.js
Backend Code:
github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/lambda-functions/childcare-database-post-function.py (Lines 28-46)

INSERT INTO activity (name, type, employee_id) VALUES
('[activity_name]', '[activity_type]', '[employee_id]');

DELETE

# When a child is deleted, the corresponding child_does_activity also gets deleted
Frontend Code:
https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/childcare-website/src/Components/Delete.js
Backend Code:
github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/lambda-functions/childcare-database-post-function.py (Lines 47-64)

```
DELETE FROM child
WHERE member_id= (user_input);
```

UPDATE

Frontend Code:
https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/childcare-website/src/Components/Delete.js
Backend Code:
github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/lambda-functions/childcare-database-post-function.py (Lines 66-84)

```
UPDATE family
SET branch_id = 'BX'
WHERE family_id = ‘FX’;
```

Selection

# Find foods that are vegetarian and takes time of 00:15:00 to prepare
Frontend Code:
https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/childcare-website/src/Components/Selection.js
Backend Code:
https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/lambda-functions/childcare-database-function.py (Lines 33-35)

```
SELECT * FROM lunch_option
WHERE (is_vegetarian = true AND time_to_prepare = “00:15:00”);
```

Projection

# View the different columns of schedule

Frontend Code:
github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/childcare-website/src/Components/ScheduleSelection.js
Backend Code:
github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/lambda-functions/childcare-database-function.py (Lines 42-87)

```
SELECT
[schedule_id/monday/tuesday/wednesday/thursday/friday/saturday/sunday/start_time/end_time]
FROM schedule;
```

Join

Frontend Code:
https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/childcare-website/src/Components/AggregationGroupBy.js
Backend Code:
https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/lambda-functions/childcare-database-function.py (Lines 125-143)

```
SELECT ct.name AS caretaker_name,
        ct.employee_id,
        ct.branch_id,
        cb.childcare_address AS branch_address
FROM caretaker ct
JOIN   childcare_branches cb
ON     ct.branch_id = cb.branch_id
WHERE ct.branch_id = 'BX';
```

Aggregation with GROUP BY

# Find the number of children in each age range
Frontend Code:

https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/childcare-website/src/Components/AggregationGroupBy.js
Backend Code:
https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/lambda-functions/childcare-database-function.py (Lines 36-38)
SELECT age, COUNT(age) AS "age_count" FROM birthday_to_age
GROUP BY age;

Aggregation with HAVING

# Search for branches where total capacity of all rooms is >= to some number

Frontend Code:
github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/childcare-website/src/Components/BranchFinder.js
Backend Code:
github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/lambda-functions/childcare-database-function.py (Lines 181-197)

SELECT SUM(room_capacity), cbr.branch_id
FROM childcare_branch_room cbr, childcare_branches cb
WHERE cbr.branch_id=cb.branch_id
GROUP BY branch_id
HAVING SUM(room_capacity) >= [TOTAL_CAPACITY];

Nested Aggregation with GROUP BY

Frontend Code:
https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/childcare-website/src/Components/AggregationGroupBy.js
Backend Code:
https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/lambda-functions/childcare-database-function.py (Lines 144-160)

```
SELECT
  f.family_id,
  AVG(EXTRACT(YEAR FROM CURRENT_DATE) - EXTRACT(YEAR FROM c.birthday)) AS
avg_child_age
FROM
  family_member f
JOIN
  child c ON f.member_id = c.member_id
GROUP BY
  f.family_id;
```

Division

# Find all the ids of the children who had participated in all the activities

Frontend Code:
https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/childcare-website/src/Components/Division.js

Backend Code:
https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_l2s4b_q5y7r_w3y3p/blob/main/lambda-functions/childcare-database-function.py (Lines 39-41)

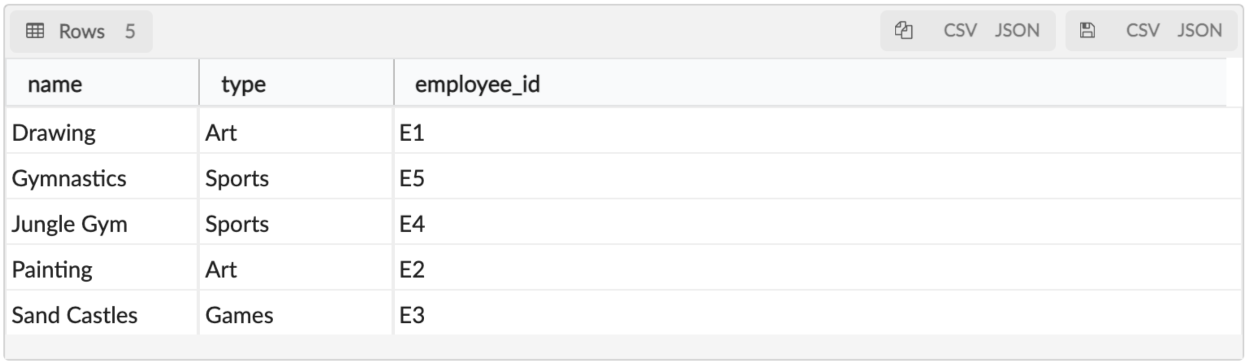
```
SELECT member_id FROM child
WHERE NOT EXISTS(SELECT name FROM activity
EXCEPT (SELECT activity_name FROM child_does_activity
WHERE child_member_id = member_id));
```

<u>5. Screenshots Demonstrating the Query Functionality using the GUI</u>

INSERT

# Add an activity based on available employee ids and user input

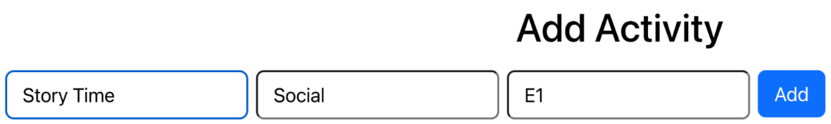
Below is a screenshot of the activity table before adding an activity. There are 5 activities in total.



Rows 5 CSV JSON CSV JSON

name	type	employee_id
Drawing	Art	E1
Gymnastics	Sports	E5
Jungle Gym	Sports	E4
Painting	Art	E2
Sand Castles	Games	E3

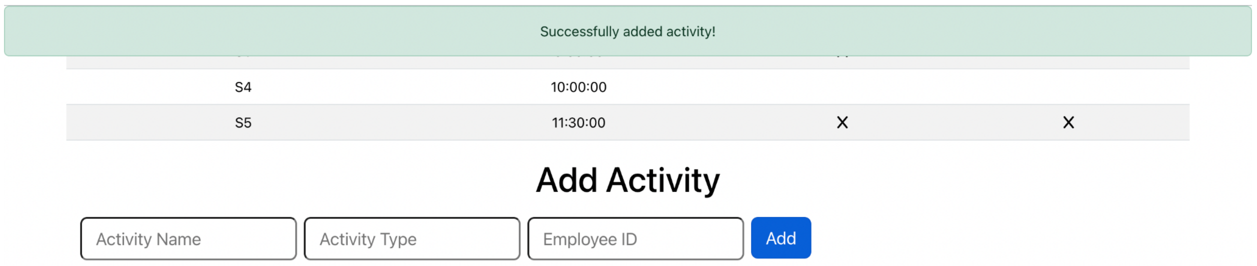
We can insert a new activity by specifying the activity name, type and employee associated with the activity.



Add Activity

Story Time	Social	E1	Add

If the insert is successful, a notification will appear at the top and the text fields will clear.



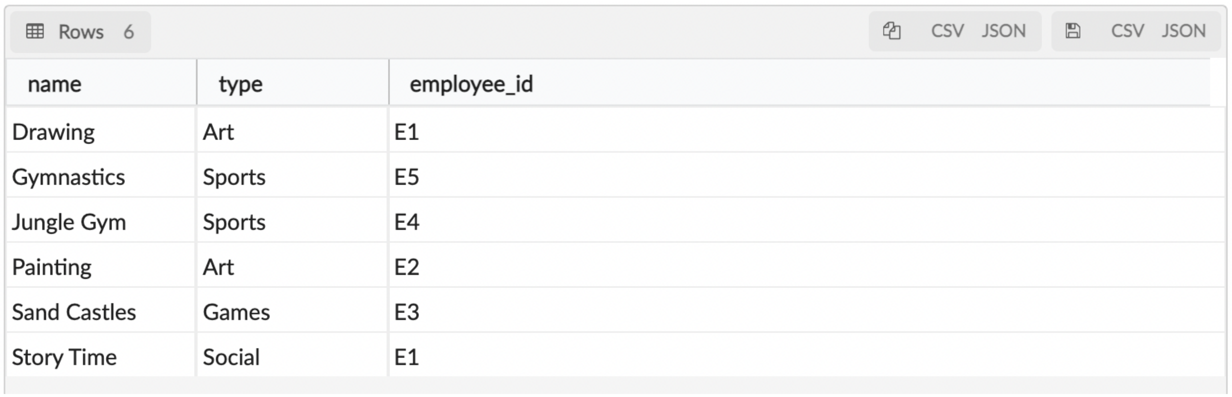
Successfully added activity!

S4	10:00:00		
S5	11:30:00	X	X

Add Activity

Activity Name	Activity Type	Employee ID	Add

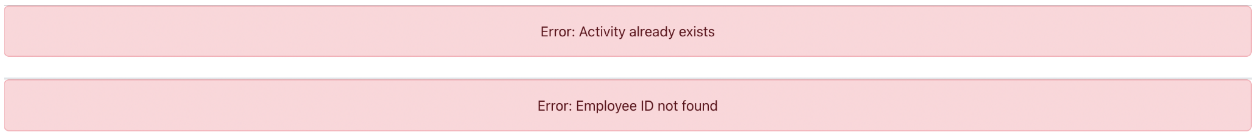
In the activity table we can now see a new activity named ‘Story Time’ with type ‘Social’ and employee ID ‘E1’.



Rows 6 CSV JSON CSV JSON

name	type	employee_id
Drawing	Art	E1
Gymnastics	Sports	E5
Jungle Gym	Sports	E4
Painting	Art	E2
Sand Castles	Games	E3
Story Time	Social	E1

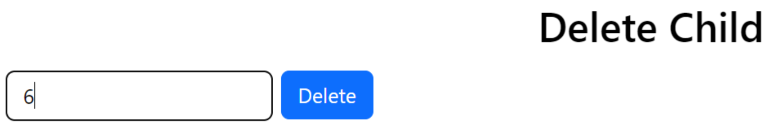
If the insert is unsuccessful a notification will appear at the top describing either a duplicate value error or a foreign key employee id not found error.


Error: Activity already exists

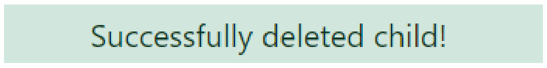
Error: Employee ID not found

DELETE

# When a child is deleted, the corresponding child_does_activity also gets deleted.
In this screenshot, we are deleting the child with member_id 6.

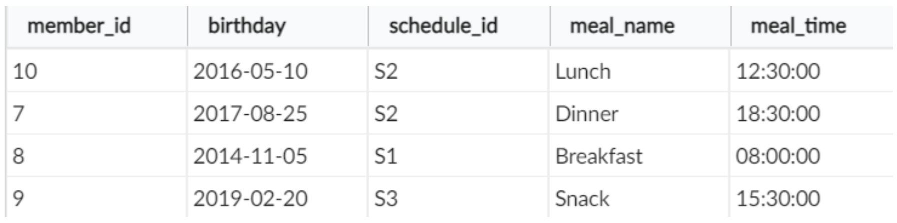


A message will be shown validating the success of the deletion on the top of the screen.


Successfully deleted child!

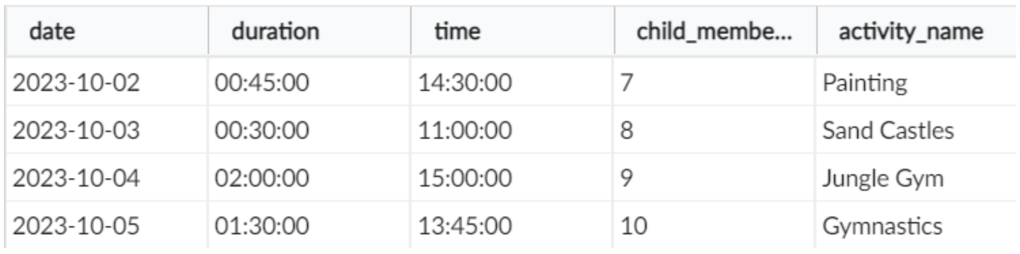
We can check that the child and the corresponding child_does_activity has also been deleted.

child:



member_id	birthday	schedule_id	meal_name	meal_time
10	2016-05-10	S2	Lunch	12:30:00
7	2017-08-25	S2	Dinner	18:30:00
8	2014-11-05	S1	Breakfast	08:00:00
9	2019-02-20	S3	Snack	15:30:00

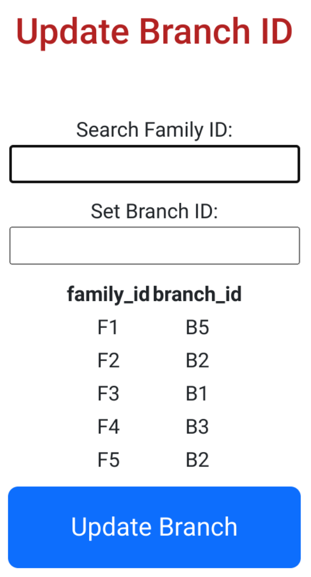
child_does_activity:



date	duration	time	child_membe...	activity_name
2023-10-02	00:45:00	14:30:00	7	Painting
2023-10-03	00:30:00	11:00:00	8	Sand Castles
2023-10-04	02:00:00	15:00:00	9	Jungle Gym
2023-10-05	01:30:00	13:45:00	10	Gymnastics

UPDATE

Based on the parameter in the family ID search box it will filter the tuples with each change. When only 1 family ID is returned, that family ID is then passed to the proper Update query that allows you to change their associated branch ID for any other branch ID that already exists upon button press.

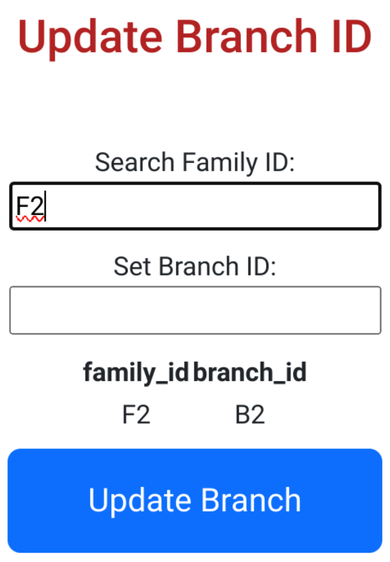


Update Branch ID

Search Family ID:

Set Branch ID:

family_id	branch_id
F1	B5
F2	B2
F3	B1
F4	B3
F5	B2

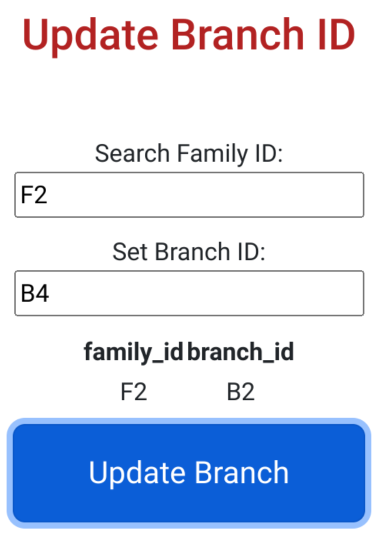


Update Branch ID

Search Family ID: F2

Set Branch ID:

family_id	branch_id
F2	B2

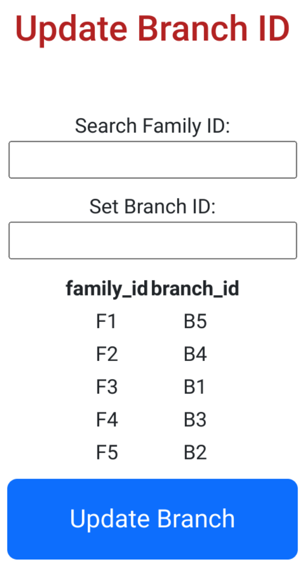


Update Branch ID

Search Family ID: F2

Set Branch ID: B4

family_id	branch_id
F2	B2



Update Branch ID

Search Family ID:

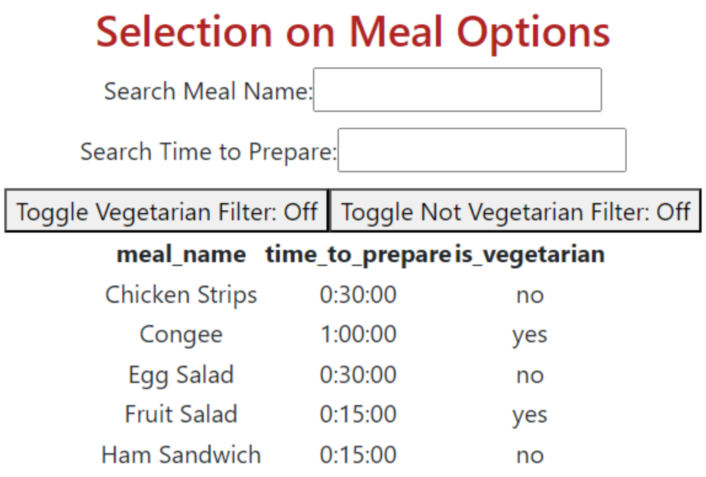
Set Branch ID:

family_id	branch_id
F1	B5
F2	B4
F3	B1
F4	B3
F5	B2

Selection

# Find foods that are vegetarian and takes time of 00:15:00 to prepare

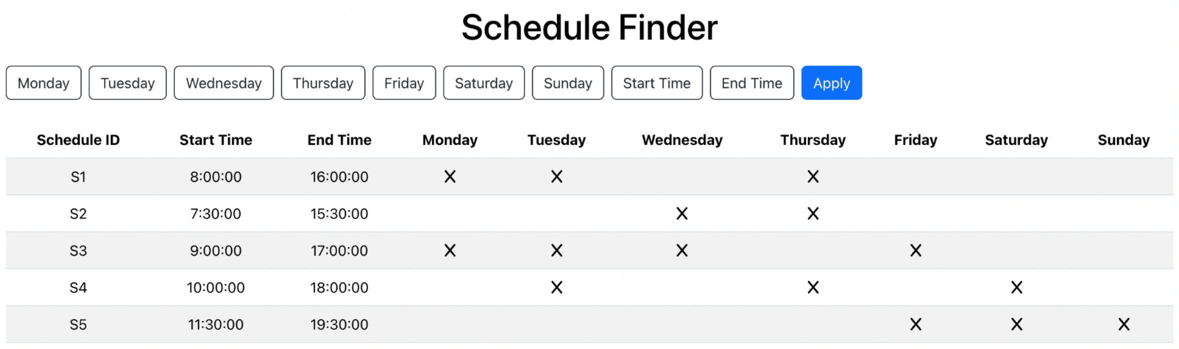
We can select those conditions by turning on the Vegetarian toggle and entering the desired preparation times.


Selection on Meal Options
Search Meal Name:
Search Time to Prepare:
Toggle Vegetarian Filter: Off
Toggle Not Vegetarian Filter: Off

meal_name	time_to_prepare	is_vegetarian
Chicken Strips	0:30:00	no
Congee	1:00:00	yes
Egg Salad	0:30:00	no
Fruit Salad	0:15:00	yes
Ham Sandwich	0:15:00	no

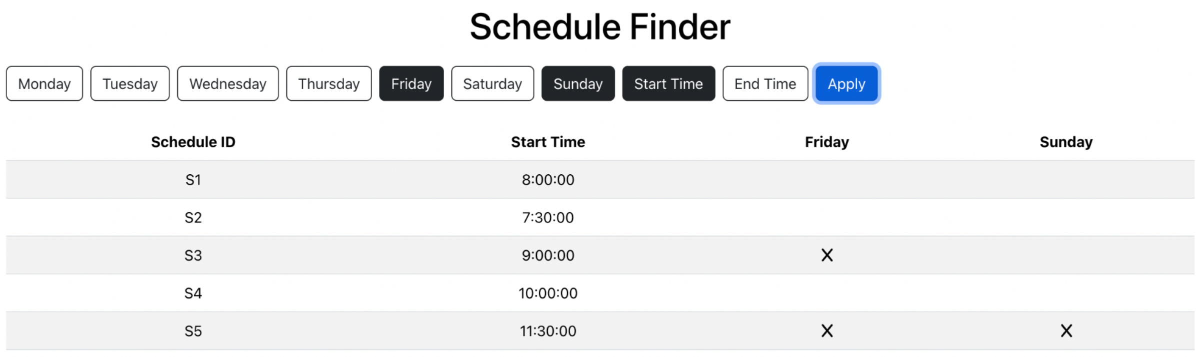
Projection

# View the different columns of schedule



Schedule ID	Start Time	End Time	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
S1	8:00:00	16:00:00	X	X		X			
S2	7:30:00	15:30:00			X	X			
S3	9:00:00	17:00:00	X	X	X		X		
S4	10:00:00	18:00:00		X		X		X	
S5	11:30:00	19:30:00					X	X	X

In this component, you are able to select the columns of the schedule table to view. If no columns are selected, all columns are shown. Each X in the table represents a true value in the database and each blank represents a false value in the database. The Schedule ID column is set to always be shown to be able to identify the rows that the table outputs.

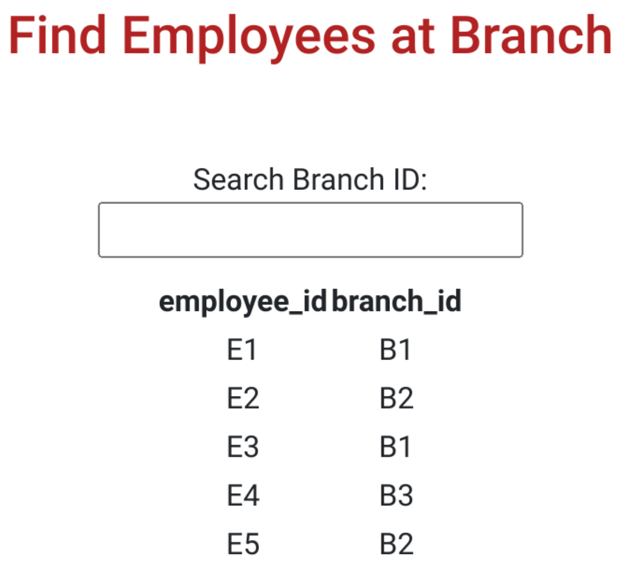


Schedule ID	Start Time	Friday	Sunday
S1	8:00:00		
S2	7:30:00		
S3	9:00:00	X	
S4	10:00:00		
S5	11:30:00	X	X

In this screenshot, we have selected the Friday, Sunday, and Start Time column. The resulting table displays these 3 columns with the Schedule ID column.

Join

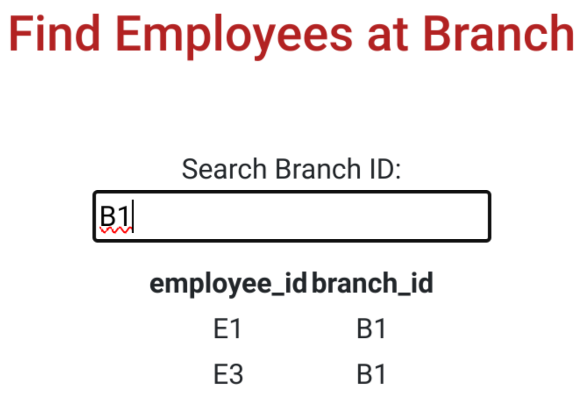
# Display all the current employees by id and their associated branch


Find Employees at Branch

Search Branch ID:

employee_id	branch_id
E1	B1
E2	B2
E3	B1
E4	B3
E5	B2

The input in the textbox is passed to the query and then displays all employees whose associated branch matches the search parameter


Find Employees at Branch

Search Branch ID:

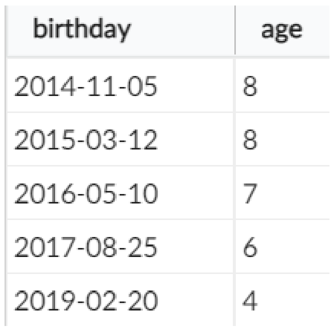
B1

employee_id	branch_id
E1	B1
E3	B1

Aggregation with GROUP BY

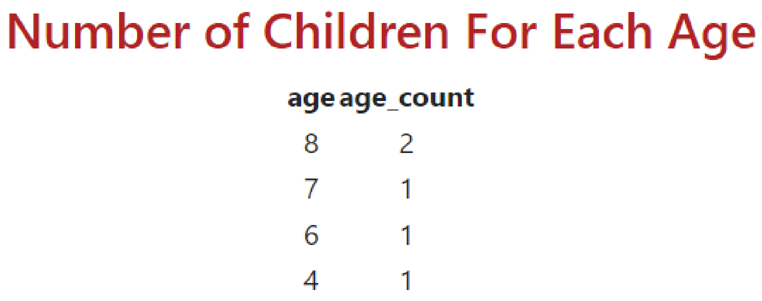
# Find the number of children in each age range

With initial data, the table birthday_to_age looks like this:



birthday	age
2014-11-05	8
2015-03-12	8
2016-05-10	7
2017-08-25	6
2019-02-20	4

After the query, the ages are grouped.



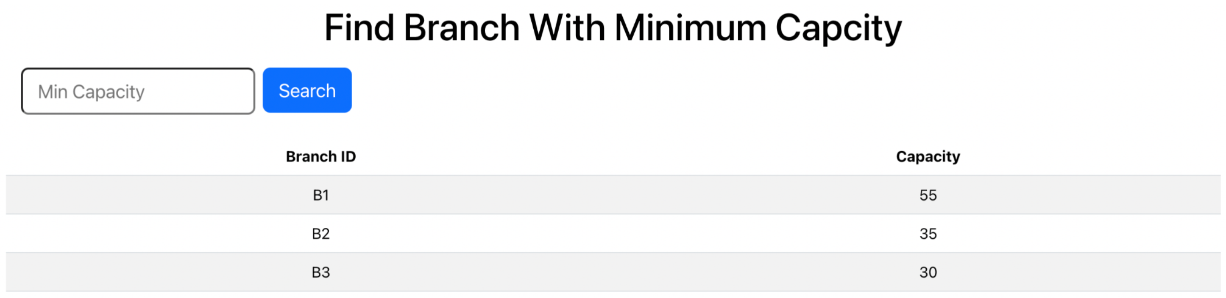
Number of Children For Each Age

age	age_count
8	2
7	1
6	1
4	1

Aggregation with HAVING

# Search for branches where total capacity of rooms is >= to some number

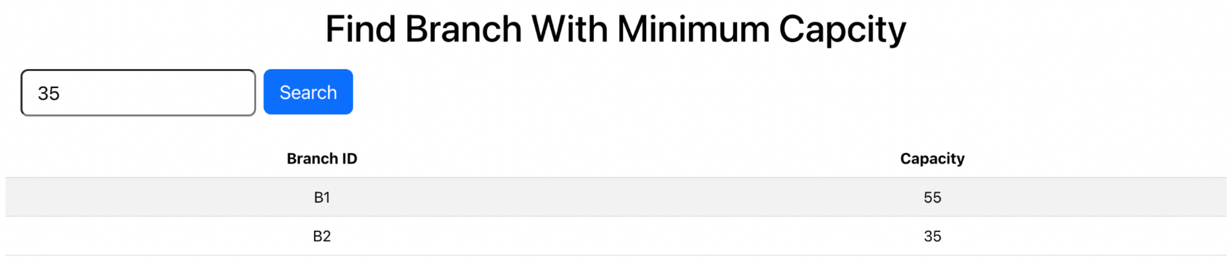
Users are able a minimum capacity and click search to search for branches where the capacity of all rooms within the branch summed up is greater than or equal to their inputted capacity. Below we can see the total capacity of all branches.



Find Branch With Minimum Capcity

Branch ID	Capacity
B1	55
B2	35
B3	30

If we input 35 in the min capacity text field and click search, the resulting table will only show the B2 and B1 branch and their maximum capacities.

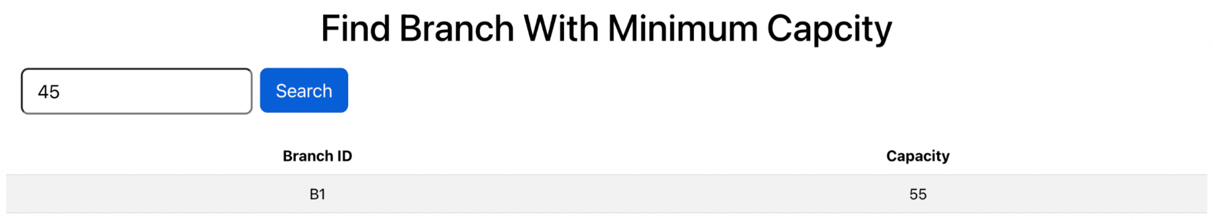


Find Branch With Minimum Capcity

35 Search

Branch ID	Capacity
B1	55
B2	35

If we input a number between 36 and 55 in the min capacity text field, the resulting table will only display the B1 branch and its corresponding capacity.



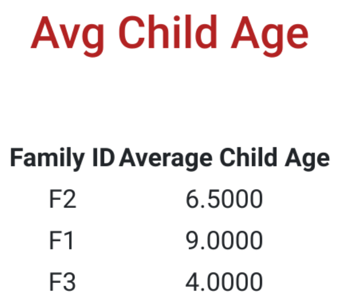
Find Branch With Minimum Capcity

45 Search

Branch ID	Capacity
B1	55

Nested Aggregation with GROUP BY

On page loading it displays the average age of all the children associated with each family id.



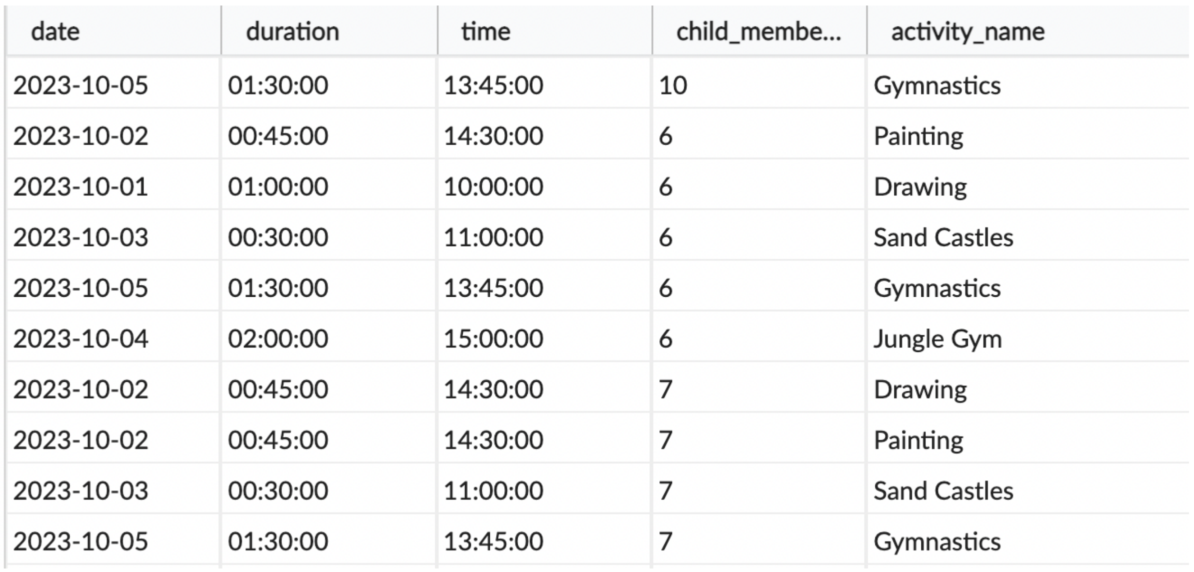
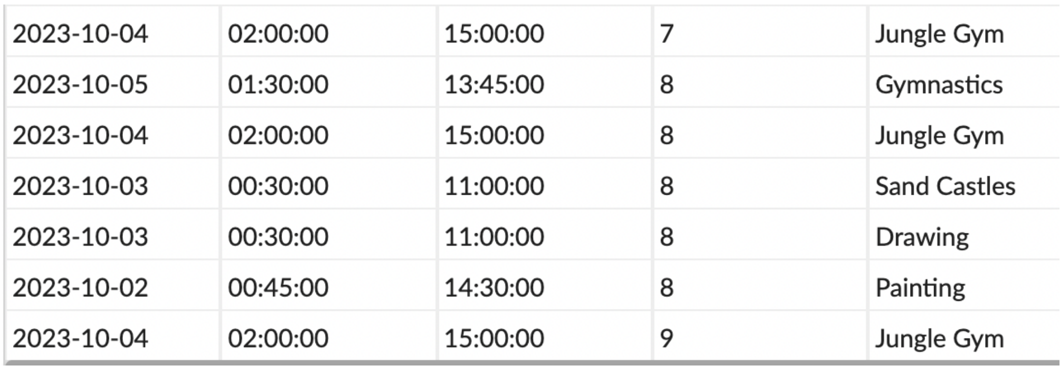
Avg Child Age

Family ID	Average Child Age
F2	6.5000
F1	9.0000
F3	4.0000

Division

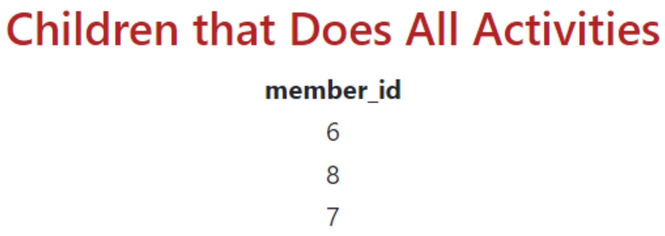
# Find all the ids of the children who had participated in all the activities

Since there are only 5 activities (Drawing, Painting, Sand Castles, Jungle Gym, and Gymnastics), in the child_does_activity table, we can see that only the children with child_member_id 6, 7, and 8 have participated in all the activities.

date	duration	time	child_membe...	activity_name
2023-10-05	01:30:00	13:45:00	10	Gymnastics
2023-10-02	00:45:00	14:30:00	6	Painting
2023-10-01	01:00:00	10:00:00	6	Drawing
2023-10-03	00:30:00	11:00:00	6	Sand Castles
2023-10-05	01:30:00	13:45:00	6	Gymnastics
2023-10-04	02:00:00	15:00:00	6	Jungle Gym
2023-10-02	00:45:00	14:30:00	7	Drawing
2023-10-02	00:45:00	14:30:00	7	Painting
2023-10-03	00:30:00	11:00:00	7	Sand Castles
2023-10-05	01:30:00	13:45:00	7	Gymnastics
2023-10-04	02:00:00	15:00:00	7	Jungle Gym
2023-10-05	01:30:00	13:45:00	8	Gymnastics
2023-10-04	02:00:00	15:00:00	8	Jungle Gym
2023-10-03	00:30:00	11:00:00	8	Sand Castles
2023-10-03	00:30:00	11:00:00	8	Drawing
2023-10-02	00:45:00	14:30:00	8	Painting
2023-10-04	02:00:00	15:00:00	9	Jungle Gym

Thus, in the division, we can see only the children with child_member_ids 6,7, and 8 are returned.



Children that Does All Activities

member_id
6
8
7