Financial Opportunity Center Database System

May 2022

Designed and presented by: Emily Liang, Elly Pham

Abstract

The Financial Opportunity Center is one of Houston's five career and personal financial service centers. FOC focuses on improving financial stability for low-to-moderate-income families and provides services across three departments: career improvement, financial education, and public benefits. Each department maintains its file-based client database, hence limiting the collaboration effort between the staff of each department. We're designing a database for Staff, Department, Funding, Funder, Service, Client Case, and Clients. The new database will enhance the collaboration between the departments and improve programs' performance.

Mission Statement

The Financial Opportunity Center database aims to keep track of client data, support FOC daily operations, facilitate collaboration between the departments, and improve program outcomes.

Mission Objectives

To maintain (add, update, and delete) data on Staff.

To maintain (add, update, and delete) data on Department.

To maintain (add, update, and delete) data on Funding.

To maintain (add, update, and delete) data on Funder.

To maintain (add, update, and delete) data on Service.

To maintain (add, update, and delete) data on Client Case.

To maintain (add, update, and delete) data on Client.

To perform searches on Staff.

To perform searches on Department.

To perform searches on Funding.

To perform searches on Funder.

To perform searches on Service.

To perform searches on Client Case.

To perform searches on Client.

To report on Staff.

To report on Departments.

To report on Funding.

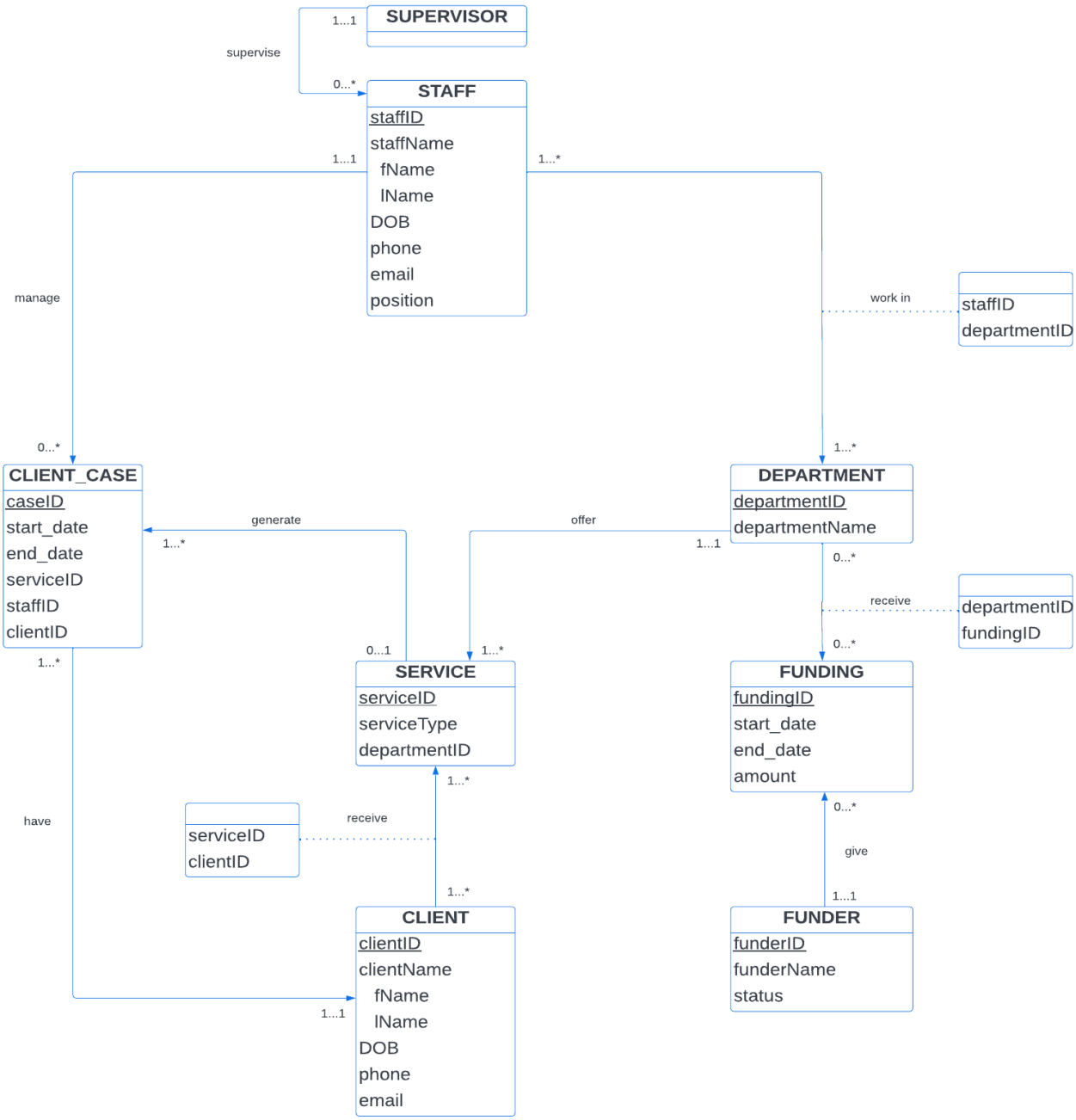
To report on Funder.

To report on Service.

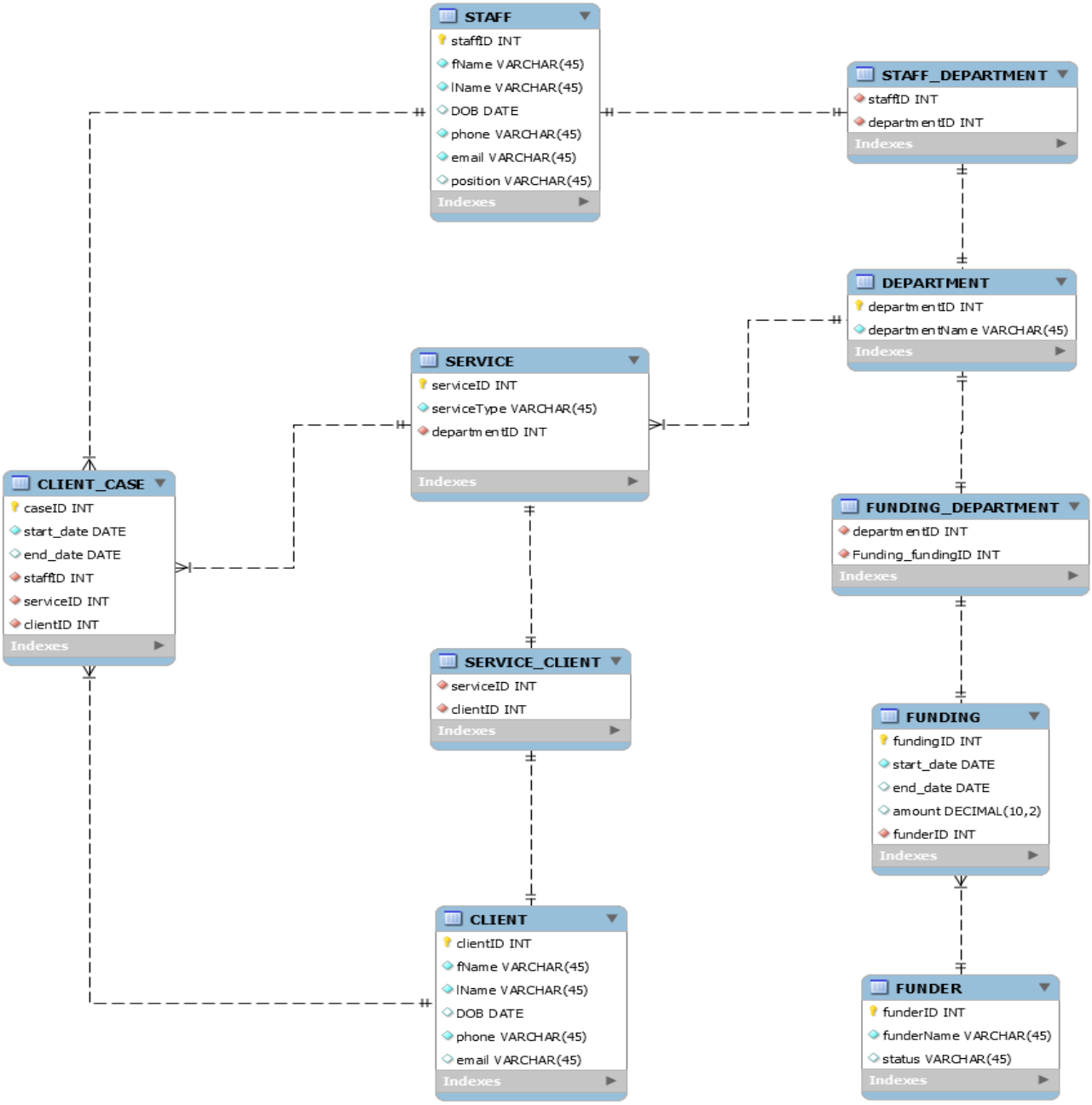
To report on Client Case.

To report on Client.

FOC ER Diagram



FOC ER Diagram in MySQL workbench



Relational Model

The following schemas describe the entities for the FOC database system. The primary keys are underlined, and the foreign keys are highlighted in blue in each schema.

Staff (staffID, fName, lName, DOB, phone, email, position)

Staff_Department(staffID, departmentID)

Department (departmentID, departmentName)

Department_Funding(departmentID, fundingID)

Funding (fundingID, start_date, end_date, amount, funderID)

Funder (funderID, funderName, status)

Service (serviceID, serviceName, departmentID)

Service_Client(serviceID, clientID)

Client (clientID, fName, lName, DOB, phone, email)

Client_Case (caseID, start_date, end_date, serviceID, staffID, clientID)

Staff Relation

Staff relations contain a record for each staff in the FOC database.

Staff (staffID, fName, lName, DOB, phone, email, position)

Key constraints: The staffID attribute is the primary key of the staff relation. Phone is the candidate key.

Referential integrity constraints: There is no foreign key in this relation.

Null constraints: The staffID, fName, lName, and phone cannot be Null.

Attribute	Domain Meaning	Data Type
staffID	All possible staff identification numbers	Integer
fName	All possible first names of staff	Varchar 45

lName	All possible last names of staff	Varchar 45
DOB	All possible birth date	Date
phone	All possible phone numbers	Varchar 45
email	All possible email address	Varchar 45
position	All possible job titles	Varchar 45

Staff Department Relationship

Staff_Department(staffID, departmentID)

Key constraints: staffID and departmentID are the primary keys of this relation.

Referential integrity constraints: staffID and departmentID are foreign key in this relation.

Null constraints: staffID and departmentID cannot be Null.

Attribute	Domain Meaning	Data Type
staffID	All possible staff identification numbers	Integer
departmentID	All possible department identification numbers	Integer

Department Relation

Department relation contains a record of the department information within FOC.

Departments (departmentID, departmentName)

Key constraints: The departmentID attribute is the primary key. There is no candidate key for Department Relation.

Referential integrity constraints: there is no foreign key in this relation.

Null constraints: The departmentID and departmentName cannot be Null.

Attribute	Domain Meaning	Data Type
departmentID	All possible department identification numbers	Integer

departmentName	All possible department names	Varchar 45

Department Funding Relationship

Funding_Department(fundingID, departmentID)

Key constraints: fundingID and departmentID are the primary keys of this relation.

Referential integrity constraints: fundingID and departmentID are foreign key in this relation.

Null constraints: fundingID and departmentID cannot be Null.

Attribute	Domain Meaning	Data Type
fundingID	All possible funding identification numbers	Integer
departmentID	All possible department identification numbers	Integer

Funding Relation

Funding Relation contains a record of funding that the department receives.

Funding (fundingID, start_date, end_date, amount, funderID)

Key constraints: The fundingID attribute is the primary key. There is no candidate key for Fundings Relation.

Referential integrity constraints: funderID is a foreign key in this relation.

Null constraints: The fundingID cannot be Null.

Attribute	Domain Meaning	Data Type
fundingID	All possible funding identification numbers	Integer
start_date	All possible dates	Date
end_date	All possible dates	Date
amount	All possible dollar amount	Decimal(10,2)
funderID	All possible funder identification numbers	Integer

Funder Relation

Funder Relation contains a record of active and inactive funder information.

Funder (funderID, funderName, status)

Key constraints: The fundersID attribute is the primary key. funderName is candidate key for Funders Relation.

Referential integrity constraints: there is no foreign key in this relation.

Null constraints: The funderID, funderName, status cannot be Null.

Attribute	Domain Meaning	Data Type
funderID	All possible funder identification numbers	Integer
funderName	All possible funder names	Varchar 20
status	Active or inactive	Varchar 10

Service Relation

Service Relation contains a record of the type of services that the department offer.

Service (serviceID, serviceName, departmentID)

Key constraints: The serviceID attribute is the primary key. serviceName ia candidate key for Services Relation.

Referential integrity constraints: departmentID is a foreign key in this relation.

Null constraints: The serviceID, serviceName cannot be Null.

Attribute	Domain Meaning	Data Type
serviceID	All possible service identification numbers	Integer
serviceName	All possible service types	Varchar 20

departmentID	All possible department identification numbers	Integer

Service Client Relationship

Service_Client(serviceID, clientID)

Key constraints: serviceID and clientID are the primary keys of this relation.

Referential integrity constraints: serviceID and clientID are foreign key in this relation.

Null constraints: serviceID and clientID cannot be Null.

Attribute	Domain Meaning	Data Type
serviceID	All possible service identification numbers	Integer
clientID	All possible client identification numbers	Integer

Clients Relation

Client Relation contains a record of FOC's client information.

Client (clientID, fName, lName, DOB, phone, email)

Key constraints: The clientID attribute is the primary key. Phone is a candidate key for Clients Relation.

Referential integrity constraints: there is no foreign key in this relation.

Null constraints: The clientID, fName, lName, phone cannot be Null.

Attribute	Domain Meaning	Data Type
clientID	All possible client identification numbers	Integer
fName	All possible first names of staff	Varchar 20
lName	All possible last names of staff	Varchar 20
DOB	All possible birth date	Date
phone	All possible phone numbers	Integer

email	All possible email address	Varchar 50

Client Case Relationship

Client Case Relation contains a record of FOC's client case.

Client_Case (caseID, start_date, end_date, serviceID, clientID, staffID)

Key constraints: The caseID attribute is the primary key. There is no candidate key for Client_Case Relation.

Referential integrity constraints: serviceID, clientID, staffID are foreign keys in this relation.

Null constraints: The caseID, start_date cannot be Null.

Attribute	Domain Meaning	Data Type
casefID	All possible case identification numbers	Integer
start_date	All possible dates	Date
end_date	All possible dates	Date
serviceID	All possible service identification numbers	Integer
clientID	All possible client identification numbers	Integer
staffID	All possible staff identification numbers	Integer

Normalization

Staff Relation

Staff (staffID, fName, lName, DOB, phone, email, position)

staffID, email -> fName, lName, DOB, position (Primary Key)

email -> phone (Transitive Dependency)

It is in the 2NF. In order to covert to 3NF, we remove the transitively dependent attribute(s) from the relation by placing the attribute(s) in a new relation along with a copy of the determinant. The result is as follows:

Staff(staffID, fName, lName, DOB, phone, email, position)

Staff_Contact(email, phone)

Staff Department Relation

Staff_Department(staffID, departmentID)

It is in 3NF.

Staff_Department(staffID, departmentID)

Department Relation

Departments (departmentID, departmentName)

It is in 3NF.

Departments (departmentID, departmentName)

Department Funding Relationship

Staff_Department(fundingID, departmentID)

It is in 3NF.

Staff_Department(fundingID, departmentID)

Funding Relation

Funding (fundingID, start_date, end_date, amount)

It is in 3NF.

Funding (fundingID, start_date, end_date, amount)

Funder Relation

Funder (funderID, funderName, status)

It is in 3NF.

Funder (funderID, funderName, status)

Service Relation

Service (serviceID, serviceName, departmentID)

It is in 3NF.

Service (serviceID, serviceName, departmentID)

Service Client Relationship

Service_Client(serviceID, clientID)

It is in 3NF.

Service_Client(serviceID, clientID)

Clients Relation

Client (clientID, fName, lName, DOB, phone, email)

clientID, email -> fName, lName, DOB

email -> phone

It is in the 2NF. In order to covert to 3NF, we remove the transitively dependent attribute(s) from the relation by placing the attribute(s) in a new relation along with a copy of the determinant. The result is as follows:

Client(clientID, fName, lName, DOB, email)
Client_Contact(email, phone)

Client Case Relation

Client_Case (caseID, start_date, end_date, serviceID, clientID, staffID)

It is in 3NF.

Client_Case (caseID, start_date, end_date, serviceID, clientID, staffID)

Use Case

Actor/User: Human Resource Manager

Use Case 1:

Use case name: Enter a new staff information

This step is done by the Human Resource Manager to enter new staff information.

Steps:

1. User goes to "Staff" table
2. User clicks "Create new staff"
3. The Database generates a staff ID
4. User enters the required fields in the Database including First Name, Last Name, DOB, Gender, Phone Number, Job Title, Start Date.
5. User clicks "Confirm" to save the information
6. New staff information is displayed

Use Case 2:

Use case name: Update staff information

This step is done by the Human Resource Manager to update the staff information.

Steps:

1. User goes to the “Staff” table
2. User enters the Staff ID or Staff Last Name

3. Prompt the user to edit the current staff information such as First Name, Last Name, Phone Number, Position.
4. All information is displayed; ask for confirmation.
5. User clicks on the “Confirm” button.

Use Case 3:

Use case name: Delete staff information

This step is done by the Human Resource Manager to delete the staff information.

Steps:

1. User goes to the “Staff” table
2. User enters the Staff ID or Staff Last Name
3. Prompt user to select the staff profile
4. User clicks “Delete”; ask for confirmation
5. User clicks on the “Confirm” button
6. The selected staff information is deleted from the database

Actor/User: Program Director

Use Case 4:

Use case name: Enter a department information

This step is done by the Program Director to enter a department information.

Steps:

1. User goes to "Department" table
2. User clicks "Create new department"
3. The Database generates a Department ID
4. User enters the required fields in the database including Department Name

5. User clicks "Confirm" to save the information
6. New department information is displayed

Use Case 5:

Use case name: Update department information

This step is done by the Program Director to update the department information.

Steps:

1. User goes to the “Department” table
2. User enters the Department ID or Department Name
3. Prompt user to edit the current department information such as Department Name
4. All information is displayed; ask for confirmation.
5. User clicks on the “Confirm” button.

Use Case 6:

Use case name: Delete department information

This step is done by Program Director to update the department information.

Steps:

1. User goes to the “Department” table
2. User enters department ID or department name
3. Prompt user to select the department profile
4. User clicks “Delete”; ask for confirmation
5. User clicks on the “Confirm” button
6. The department information is deleted from the database

Use Case 7:

Use case name: Enter funding information

This step is done by the Program Director to enter new funding information.

Steps:

1. User goes to "Funding" table
2. User clicks "Create new funding"
3. The database generates a Funding ID
4. User enters the required fields in the database including date start, date end, funding amount
5. User clicks "Confirm" to save the information
6. New funding information is displayed

Use Case 8:

Use case name: Update funding information

This step is done by the Program Director to update funding information.

Steps:

1. User goes to the “Department” table
2. User enters department ID or funding Name
3. Prompt user to edit funding information such as department name
4. All information is displayed; ask for confirmation
5. User clicks on the “Confirm” button

Use Case 9:

Use case name: Delete funding information

This step is done by the Program Director to delete the current funding information.

Steps:

1. User goes to the “Funding” table
2. User enters the funding ID or funding name
3. Prompt user to select the funding profile
4. User clicks “Delete”; ask for confirmation

5. User clicks on the “Confirm” button
6. The Funding information is deleted from the database

Use Case 10:

Use case name: Enter funder information

This step is done by the Program Director to enter funder information.

Steps:

1. User goes to "Funder" table
2. User clicks "Create new funder"
3. The database generates a funder ID
4. User enters the required fields in the database including name
5. User clicks "Confirm" to save the information
6. New funder information is displayed

Use Case 11:

Use case name: Update funder information

This step is done by the Program Director to update the funder information.

Steps:

1. User goes to the “Funder” table
2. User enters the funder ID or funder name
3. Prompt user to edit the current funder information such as Funder Name
4. All information is displayed; ask for confirmation.
5. User clicks on the “Confirm” button.

Use Case 12:

Use case name: Delete funder information

This step is done by the Program Director to delete the funder information.

Steps:

1. User goes to the “Funder” table
2. User enters the Funder ID or Funder Name
3. Prompt user to delete the funder information
4. User clicks “Delete”; ask for confirmation
5. User clicks on the “Confirm” button
6. The Funder information is deleted from the database

Actor/User: Data Specialist

Use Case 13:

Use case name: Enter service information

This step is done by the Data Specialist to enter service information.

Steps:

1. User goes to "Services" table
2. User clicks "Create new services"
3. The database generates a new services ID
4. User enters the required fields in the Database including Name
5. User clicks "Confirm" to save the information
6. New services information is displayed

Use Case 14:

Use case name: Update service information

This step is done by the Data Specialist to update service information.

Steps:

1. User goes to the “Services” table
2. User enters the service ID or service Name
3. Prompt user to edit service information such as service name

4. All information is displayed; ask for confirmation.
5. User clicks on the “Confirm” button.

Use Case 15:

Use case name: Delete service information

This step is done by the Data Specialist to delete service information.

Steps:

1. User goes to the “Services” table
2. User enters the services ID or service name
3. Prompt user to edit service information
4. User clicks “Delete”; ask for confirmation
5. User clicks on the “Confirm” button
6. The Services information is deleted from the database

Use Case 16:

Use case name: Enter client information

This step is done by the Data Specialist to enter client information.

Steps:

1. User goes to "Services" table
2. User clicks "Create new client”
3. The Database generates a client ID
4. User enters the required fields in the Database including fName, lName, DOB, Gender, Phone, Email
5. User clicks "Confirm" to save the information
6. New client information is displayed

Use Case 17:

Use case name: Update client information

This step is done by the Data Specialist to update client information.

Steps:

1. User goes to the “Client” table
2. User enters the client ID or client name
3. Prompt user to edit the current client information such as Client Name
4. All information is displayed; ask for confirmation
5. User clicks on the “Confirm” button

Use Case 18:

Use case name: Delete client information

This step is done by the Data Specialist to delete client information.

Steps:

1. User goes to the “Client” table
2. User enters the client ID or client name
3. User clicks “Delete”; ask for confirmation
4. User clicks on the “Confirm” button
5. The Client information is deleted from the database

Use Case 19:

Use case name: Enter new case information

This step is done by the Data Specialist to enter new case information.

Steps:

1. User goes to "Services" table
2. User clicks "Create new case”
3. The database generates a case ID

4. User enters the required fields in the Database including fName, lName, DOB, Gender, Phone, Email
5. User clicks "Confirm" to save the information
6. New case information is displayed

Use Case 20:

Use case name: Update new case information

This step is done by the Data Specialist to update new case information.

Steps:

1. User goes to the “Case” table
2. User enters the Case ID or Case Name
3. Prompt user to edit the current case information such as Case Name
4. All information is displayed; ask for confirmation
5. User clicks on the “Confirm” button

Use Case 21:

Use case name: Delete case information

This step is done by the Data Specialist to delete case information.

Steps:

1. User goes to the “Case” table
2. User enters the case ID or case Name
3. Prompt user to select case information
4. User clicks “Delete”; ask for confirmation
5. User clicks on the “Confirm” button
6. The case information is deleted from the database

Actor/User: Program Analyst

Use Case 22:

Use case name: Generate a report for total number of clients each department serviced in a given time

This step is done by the Program Analyst to generate the report using join function between **client_case**, **service** and **department** and group by department.

Steps:

1. User enters “Join relations”
2. The system displays the relationship between the entities
3. User enters the condition for filtering
4. User selects the attributes
5. User enters group by department
6. The system generates the report including departmentName, serviceType, and number of client cases

Use Case 23:

Use case name: Generate a report for the number of client cases that each staff serviced in a given time.

This step is done by the Program Analyst to generate the report using join function between **Client_Case** and **Staff** and group by staff name

Steps:

1. User enters “Join relations”
2. The system displays the relationship between the entities
3. User enters the condition for filtering
4. User selects the attributes
5. User enters group by staff name
6. The system generates the report including staffName

Use Case 24:

Use case name: Generate a report for the minimum, maximum number of cases one client received in a given time

This step is done by the Program Analyst to generate the report from **Client_Case** and group by clientID.

Steps:

1. User selects the attributes
2. User clicks the aggregate function such as minimum, maximum, average
3. User enters the condition for filtering
4. User enters group by client ID
5. The system generates the report including clientID, minimum, maximum number of services

Use Case 25:

Use case name: Generate a report for number of new grants the departments received in a given period

This step is done by the Program Analyst to generate the report using join function between **Funding**, and **Department, Funding_Department** and group by department.

Steps:

1. User enters “Join relations”
2. The system displays the relationship between the entities
3. User enters the condition for filtering
4. User selects the attributes
5. User enters group by Department
6. The system generates the report including departmentName, the time interval, and the number of funding

Use Case 26:

Use case name: Generate a report for the grant amount funded by funder in a given time and sort by the grant amount.

This step is done by the Program Analyst to generate the report using join function between **funding**, and **funder, funding_department, Service,** and group by each service type.

Steps:

1. User enters “Join relations”
2. The system displays the relationship between the entities
3. User enters the condition for filtering
4. User selects the attributes of each entity
5. User enters group by service type
6. User click the “Sort” button for the selected attribute
7. The system generates the report including service type, funder name, funding amount

Use Case 27:

Use case name: Generate a report of the number of clients serviced by service types in a given time.

This step is done by the Program Analyst to generate the report using join function between **service**, and **client_case** and group by servicetype.

Steps:

1. User enters “Join relations”
2. The system displays the relationship between the entities
3. User enters the condition for filtering
4. User selects the attributes
5. User enters group by serviceType
6. User click the “Sort” button by the total of number client cases
7. The system generates the report including serviceType, total number of client cases

Use Case 28:

Use case name: Generate a report for total number of staffs of each department

This step is done by the Program Analyst to generate the report using join function between **Staff**, **Staff_Department** and **Department** and group by department.

Steps:

1. User enters "Join relations"
2. The system displays the relationship between the entities
3. User selects the attributes of each entity
4. User enters group by Department Name
5. The system generates the report including department name, total number of staffs

Use Case Realization - SQL Statements

UC1

```sql
-- Enter new staff information
INSERT INTO STAFF(fName, lName, DOB, phone, email, position) VALUES
('Pham', 'El', '1989-12-01','832-123-4113','elpham@foc.org','Program Analyst');
```

UC2

```sql
-- Update staff information
UPDATE STAFF
SET phone = "832-789-6566"
WHERE fName = "El";
```

UC3

```sql
-- Delete staff information
DELETE FROM STAFF
WHERE fName = "El"
AND lName = "Pham";
```

UC4

```sql
-- Enter new staff information
INSERT INTO STAFF(fName, lName, DOB, phone, email, position) VALUES
('Pham', 'El', '1989-12-01','832-123-4113','elpham@foc.org','Program Analyst');
```

UC5

```sql
-- Update department information
UPDATE DEPARTMENT
SET departmentName = "Benefits Enrollment Center"
WHERE departmentName = "Public Benefits";
```

UC6

```sql
-- Delete department information
DELETE FROM DEPARTMENT
WHERE departmentName = "Senior Job";
```

UC7

```sql
-- Enter a new funding information
INSERT INTO FUNDING(start_date, end_date, amount, funderID) VALUES
('2018-01-03', '2021-04-18', '416421.00', '2');
```

UC8

```sql
-- Update a new funding information
UPDATE FUNDING
SET amount = "416421.00"
WHERE fundingID = 8;
```

UC9

```sql
-- Delete funding information
DELETE FROM FUNDING
WHERE fundingID = 8;
```

UC10

```sql
-- Enter a new funder information
INSERT INTO FUNDER (funderName, status) VALUES
('Robo AI', 'Inactive');
```

UC11

```sql
-- Update a new funder information
UPDATE FUNDER
SET status = "Active"
WHERE funderName = "robo ai";
```

UC12

```sql
-- Delete funder information
DELETE FROM FUNDER
WHERE funderName = "Robo AI";
```

UC13

```sql
-- Enter a new service information
INSERT INTO SERVICE (serviceID, serviceType, departmentID) VALUES
(1011,'Truck Driver Training', 1);
```

UC14

```sql
-- Update a new service information
UPDATE SERVICE
SET serviceType = "Driving Class"
WHERE serviceType = "Truck Driver Training";
```

UC15

```sql
-- Delete the current service information
DELETE FROM SERVICE
WHERE serviceType = "Driving Class";
```

UC16

```
-- Enter a new client information
INSERT INTO CLIENT(fName, lName, DOB, phone, email) VALUES
('Smith', 'Amith', '1976-01-09','732-980-5791', "smith@aoc.com");
```

UC17

```
-- Update a new client information
UPDATE CLIENT
SET email = "smith.amith@gmail.com"
WHERE fName = "Smith"
AND lName = "Amith";
```

UC18

```
-- Delete the current client information
DELETE FROM CLIENT
WHERE fName = "Smith"
AND lName = "Amith";
```

UC19

```
-- Enter a new client case information
INSERT INTO CLIENT_CASE(start_date, end_date, staffID, serviceID, clientID) VALUES
('2021-08-26', '2022-04-01', 2, 1012, 123);
```

UC20

```
-- Update a new client case information
UPDATE CLIENT_CASE
SET end_date = "2022-04-16"
WHERE caseID = 511;
```

UC21

```
-- Delete the existing client case information
DELETE FROM CLIENT_CASE
WHERE caseID = 511;
```

UC22

```
-- Generate a report for the total number of clients each
-- department serves in a given time
DROP PROCEDURE tol_client_per_department;
delimiter $$
CREATE PROCEDURE tol_client_per_department (report_start_date date, report_end_date date)
BEGIN
SELECT report_start_date, report_end_date, d.departmentName, count(cc.caseID) as total_clients
FROM department d
INNER JOIN service s ON d.departmentID = s.departmentID
INNER JOIN CLIENT_CASE cc ON cc.serviceID = s.serviceID
WHERE cc.start_date BETWEEN report_start_date AND report_end_date
GROUP BY d.departmentName;
END$$
delimiter $$

CALL tol_client_per_department("2021-01-01",  "2021-03-31");
```

00% 1:116

Result Grid Filter Rows: Search Export:

report_start_date	report_end_date	departmentName	total_clients
2021-01-01	2021-03-31	Public Benefits	1
2021-01-01	2021-03-31	Financial Opportunity	1

UC23

```
-- Generate a report for the number of client cases each staff
-- provided in a given time
DROP PROCEDURE tol_client_per_staff;
delimiter $$
CREATE PROCEDURE tol_client_per_staff (report_start_date date, report_end_date date)
BEGIN
SELECT report_start_date, report_end_date, s.fName, s.lName, count(c.caseID)
FROM staff s
INNER JOIN client_case c ON s.staffID = c.staffID
WHERE c.start_date BETWEEN report_start_date AND report_end_date
GROUP BY s.fName, s.lName;
END$$
delimiter $$;

CALL tol_client_per_staff ("2021-01-01",  "2021-12-31");
```

100% 1:132

Result Grid Filter Rows: Search Export:

report_start_date	report_end_date	fName	lName	count(c.caseID)
2021-01-01	2021-12-31	Kev	In	2
2021-01-01	2021-12-31	Siv	Ian	1
2021-01-01	2021-12-31	Cin	Dy	2

UC24

```sql
-- Generate a report for the minimum, the maximum,
-- the average number of services provided in a given time.
SELECT max(total) max_services, min(total) min_services, avg(total) avg_services
FROM (SELECT count(caseID) AS total
FROM Client_Case
WHERE start_date BETWEEN "2021-01-01" AND "2021-12-31"
GROUP BY clientID) table1;
```

100% 1:141

Result Grid Filter Rows: Search Export:

max_services	min_services	avg_services
2	1	1.2500

UC25

```sql
-- Generate a report for the total number of grants receives by the department and
-- total amount in a given time period.
SELECT d.departmentName, count(f.fundingID), sum(f.amount)
FROM department d
INNER JOIN funding_department fd ON d.departmentID = fd.departmentID
INNER JOIN funding f ON f.fundingID = fd.fundingID
WHERE start_date BETWEEN "2021-01-01" AND "2021-12-31"
GROUP BY departmentName;
```

100% 1:149

Result Grid Filter Rows: Search Export:

departmentName	count(f.fundingI...	sum(f.amount)
Public Benefits	1	3923456.78

UC26

```sql
-- Generate a report of the amount of funding group by each funder in a given time period
-- and sort by the grant amount.
SELECT f.funderName, fg.start_date, fg.end_date, sum(fg.amount)
FROM funder f
INNER JOIN funding fg ON f.funderID = fg.funderID
WHERE DATE(fg.start_date) > "2020-01-03"
GROUP BY f.funderName, fg.start_date, fg.end_date
ORDER BY sum(fg.amount);
```

00% 1:161

Result Grid Filter Rows: Search Export:

funderName	start_date	end_date	sum(fg.amount)
Napka	2020-01-23	NULL	2923456.78
Baker Cake	2021-01-06	NULL	3923456.78

UC27

```sql
-- Generate a report of the service types with a higher number of clients in a given time period
SELECT s.serviceType, count(cc.caseID)
FROM Service s
INNER JOIN Client_Case cc ON s.serviceID = cc.serviceID
WHERE YEAR(cc.start_date) >  "2005"
GROUP BY s.serviceType
ORDER BY count(cc.caseID) DESC;
```

00% 1:167

Result Grid Filter Rows: Search Export:

serviceType	count(cc.caseID)
Rent	2
Utility	1
Job	1
Resume	1
Training	1
Informational	1
Education	1
Credit	1

UC29

```
-- Total number of staff per department
SELECT  d.departmentName, count(s.staffID)
FROM staff s
INNER JOIN staff_department sd ON s.staffID = sd.staffID
INNER JOIN department d ON d.departmentID = sd.departmentID
GROUP BY d.departmentName;
```

100% 1:258

Result Grid Filter Rows: Search Export:

departmentName	count(s.staffI...
Financial Opportunity	6
Public Benefits	3
Human Resource	1

Relation in MySQL

Client

```
SELECT * FROM client;
```

100% 1:2

Result Grid Filter Rows: Search Edit: Export/Import:

clientID	fName	lName	DOB	phone	email
100	Abe	Laa	1938-11-01	733-440-5517	NULL
101	Bbe	Lab	1948-01-22	712-245-5047	NULL
102	Cbe	Lac	1958-07-01	735-408-1567	1kay@gmail.com
103	Dbe	Lad	1988-12-17	732-845-8568	o5y12@gmail.com
104	Ebe	Lae	NULL	732-445-4567	NULL
105	Fbe	Laf	1973-01-05	792-445-5567	NULL
106	Gbe	Lag	1962-03-14	332-345-3564	23kay55@gmail.com
107	Hbe	Lah	NULL	732-445-5560	erkay@yahoo.com
108	Ibe	Lai	2001-09-12	632-245-0560	4kay@yahoo.com
109	Jbe	Laj	1995-02-08	735-545-5261	NULL
NULL	NULL	NULL	NULL	NULL	NULL

Client_Case

```
SELECT * FROM client_case;
```

100% 1:4

Result Grid Filter Rows: Search Edit: Export/Import:

caseID	start_date	end_date	staffID	serviceID	clientID
510	2020-01-01	2021-12-18	5	1001	100
512	2022-01-01	NULL	5	1003	102
513	2021-01-01	NULL	7	1003	103
514	2018-01-01	2022-04-18	5	1004	104
515	2020-07-01	NULL	5	1005	105
516	2020-08-11	NULL	7	1006	106
517	2021-11-12	NULL	7	1007	107
518	2021-01-01	2021-11-28	5	1008	108
519	2022-02-01	NULL	5	1009	109
520	2005-01-01	2021-05-18	7	1010	103
521	1997-01-01	2005-10-10	5	1005	108
522	2021-08-26	2022-04-01	2	1012	123
NULL	NULL	NULL	NULL	NULL	NULL

Department

```
SELECT * FROM department;
```

100% 1:6

Result Grid Filter Rows: Search Edit: Export/Import:

departmentID	departmentName
1	Financial Opportunity
2	Public Benefits
3	Human Resource
NULL	NULL

Funder

```
7 •  SELECT * FROM funder;
8
9
```

100% 1:8

Result Grid Filter Rows: Search Edit: Export/Import:

funderID	funderName	status
1	United Sky	Active
2	Harris Love	Active
3	Baker Cake	Active
4	Napka	Active
5	Oil Llc	Inactive
NULL	NULL	NULL

Funding

```
9 •  SELECT * FROM funding;
10
11
```

100% 1:10

Result Grid Filter Rows: Search Edit: Export/Import:

fundingID	start_date	end_date	amount	funderID
1	1998-01-03	2022-04-18	923456.78	1
2	2008-05-03	2021-04-18	1923456.78	1
3	2018-11-03	NULL	11923456.78	3
4	2020-01-23	NULL	2923456.78	4
5	1998-01-13	2020-01-03	3923456.78	5
6	2015-01-10	NULL	21923456.78	2
7	2021-01-06	NULL	3923456.78	3
NULL	NULL	NULL	NULL	NULL

Service

```
11 •  SELECT * FROM service;
12
13
```

100% 1:12

Result Grid Filter Rows: Search Edit: Export/Import:

serviceID	serviceType	departmentID
1001	Utility	2
1002	Food	2
1003	Rent	2
1004	Job	1
1005	Resume	1
1006	Training	1
1007	Informational	1
1008	Education	1
1009	Credit	1
1010	Emergency Assistance	2
NULL	NULL	NULL

Service_Client

```
13 •  SELECT * FROM service_client;
14
```

100% 29:13

Result Grid Filter Rows: Search Export:

serviceID	clientID
1001	102
1002	102
1003	102
1009	100
1010	101
1001	102
1006	103
1007	102
1005	106
1010	102
1007	107
1004	109
1004	108
1001	108
1006	103
1008	105
1002	105
1006	101

Staff

```
15 •  SELECT * FROM staff;
16
```

100% 20:15

Result Grid Filter Rows: Search Edit: Export/Import:

	stfName	lName	DOB	phone	email	position
1	Pama	La	1988-01-01	832-445-5567	pama@foc.org	Director
2	Cin	Dy	1983-07-01	832-445-5564	cin@foc.org	Data Specialist
3	He	Len	1968-11-01	832-445-5569	he@foc.org	Program Analyst
4	Rash	Mi	1988-01-01	832-445-5367	rash@foc.org	HR Manager
5	Siv	Ian	1998-01-11	832-445-5467	Siv@foc.org	Case Manager
6	Val	Rie	1983-12-01	832-445-5264	val@foc.org	Program Coordinator
7	Kev	In	1968-01-12	832-445-3367	kev@foc.org	Case Manager
8	Derek	Ho	1993-01-21	832-445-1267	derek@foc.org	FOC Program Manager
9	Lulu	Lu	1983-11-21	832-445-6267	lulu@foc.org	Program Coordinator
1	Tao	Yue	1958-01-21	832-445-1260	tao@foc.org	CEAP Program Manager
1	Pham	El	1989-12-01	832-123-4113	elpham@foc....	Program Analyst
N	NULL	NULL	NULL	NULL	NULL	NULL

Staff_Department

```
17 •  SELECT * FROM staff_department;
18
19
```

100% 1:18

Result Grid Filter Rows: Search Export:

staffID	departmentID
1	1
2	1
3	1
4	3
5	2
6	1
7	2
8	1
9	1
10	2

Test plan and records

Use Case	Input Data	Expected Output	Actual Output	Result
1	**INSERT INTO** staff(fName, lName, DOB, phone, email, position) **VALUES** (‘Pham’, ‘El’, ‘1989-12-01’, ‘832-123-4113’, ‘elpham@foc.org’, ‘Program Analyst');	Inserted a new staff information into staff table; staffID was auto incremented. New staff with fName ‘El’ was added to ‘Program Analyst’ position	A staff with fName ‘El’ was added to ‘Program Analyst’ position	Passed
2	**UPDATE** staff **SET** phone = "832-789-6566" **WHERE** fName = ‘El’;	Updated the phone number of staff that has fName as ‘El’ to "832-789-6566" in the staff table	The phone number of staff with fName ‘El’ was updated to ‘832-789-6566’	Passed
3	**DELETE FROM** staff **WHERE** fName = ‘El’ **AND** lName = ‘Pham’;	Deleted staff information from staff table where the fName of the staff is ‘El’ and lName is ‘Pham’	Staff with fName ‘El’ and lName ‘Pham’ was deleted and no longer found in the staff table	Passed

4	**INSERT INTO** department(departmentName) **VALUES** (‘Senior Job’);	Inserted a new department information into department table; departmentID was auto incremented. A new ‘Senior Job’ department was added	A department with departmentID 4 was added to ‘Senior Job’	Passed
5	**UPDATE** department **SET** departmentName = ‘Benefits Enrollment Center’ **WHERE** departmentName = ‘Public Benefits’;	Updated the existing departmentName to ‘Benefits Enrollment Center’ in the department table	A departmentName was updated to ‘Benefits Enrollment Center’	Passed
6	**DELETE FROM** department **WHERE** departmentName = ‘Senior Job’;	Deleted department information from department table where the departmentName is ‘Senior Job’	Department with the departmentName of ‘Senior Job’ was deleted and no longer found in the department table	Passed
7	**INSERT INTO** funding(start_date, end_date, amount, funderID) **VALUES** ('2018-01-03', '2021-04-18', '416421.00', '2');	Inserted a new funding information into funding table; fundingID was auto incremented. New funding with the amount of '416421.00' was added to funderID of 2	A funding with fundingID of 8 was added to funderID of 2	Passed
8	**UPDATE** funding **SET** amount = ‘416421.00’ **WHERE** fundingID = 8;	Inserted a new funding information into funding table; fundingID was auto incremented. New funding with the amount of '416421.00' was added to funderID of 2	A funding with fundingID of 8 was added to funderID of 2	Passed

9	**DELETE FROM** funding **WHERE** fundingID = 8;	Deleted funding information from funding table where the fundingID is 8	A funding with the fundingID of 8 was deleted and no longer found in the funding table	Passed
10	**INSERT INTO** funder(funderName, status) **VALUES** ('Robo AI', 'Inactive');	Inserted a new funder information into funder table; fundingID was auto incremented. New funder ‘Robo AI’ was added to ‘Inactive’ status	A funder with funderName 'Robo AI' was added to ‘Inactive’ status	Passed
11	**UPDATE** funder **SET** status = ‘Active’ **WHERE** funderName = ‘Robo AI’;	Updated the funder status to ‘Active’ in the funding table where the funderName is ‘Robo AI’	A funder status with funderName ‘Robo AI’ was updated to ‘Active’	Passed

12	**DELETE FROM** funder **WHERE** funderName = ‘Robo AI’;	Deleted funder information from funder table where the funderName is ‘Robo AI’	A funder with the funderName of ‘Robo AI’ was deleted and no longer found in the funder table	Passed
13	**INSERT INTO** service (serviceID, serviceType, departmentID) **VALUES** (1011,'Truck Driver Training', 1);	Inserted a new service information with serviceID of 1011 into service table. ServiceID of 1011 was added to departmentID of 1	A serviceID 1011 was added to departmentID of 1	Passed
14	**UPDATE** service SET serviceType = ‘Driving Class’ **WHERE** serviceType = ‘Truck Driver Training’;	Updated the serviceType to ‘Driving Class’ in the service table where the serviceType is ‘Truck Driver Training’	A service type with serviceID 1011 was updated to ‘Driving Class’	Passed
15	**DELETE FROM** service **WHERE** serviceType = ‘Driving Class’;	Deleted service information from service table where the serviceType is ‘Driving Class’	A service with the serviceType of ‘Driving Class’ was deleted and no longer found in the service table	Passed
16	**INSERT INTO** client(fName, lName, DOB, phone, email) **VALUES** ('Smith', 'Amith', '1976-01-09','732-980-5791', ‘smith@aoc.com’);	Inserted a new client information into client table; clientID was auto incremented. New staff with fName 'Smith' and lName ‘Amith’ was added to client table	A staff with fName ‘ Smith’, lName ‘Amith’ with email ‘smith@aoc.com’ was added	Passed

17	**UPDATE** client **SET** email = ‘smith.amith@gmail.com’ WHERE fName = ‘Smith’ AND lName = ‘Amith’;	Updated the client’s email with fName "Smith" and lName ‘Amith’ to ‘smith.amith@gmail.com’ in the client table	The corresponding email of fName ‘Smith’ and lName ‘Amith’ was updated to ‘smith.amith@gmail.com’	Passed
18	**DELETE FROM** client **WHERE** fName = ‘Smith’ **AND** lName = ‘Amith’;	Deleted client information from client table where the fName of the client is ‘Smith’ and lName is ‘Amith’	Staff with fName ‘ Smith’ and lName ‘Amith’ was deleted and no longer found in the client table	Passed

19	**INSERT INTO** client_case(start_date, end_date, staffID, serviceID, clientID) **VALUES** ('2021-08-26', '2022-04-01', 2, 1012, 123);	Inserted a new client_case information into client_case table; caseID was auto incremented. A new caseID with start date ‘2021-08-26' was added to serviceID 1012	A caseID with start date ‘2021-08-26' was added to serviceID 1012	
20	**UPDATE** client_case **SET** end_date = "2022-04-16" **WHERE** caseID = 511;	Updated the end_date with caseID 511 to "2022-04-16" in client_case table	The corresponding end_date of caseID 511 was updated to "2022-04-16"	
21	**DELETE FROM** client_case **WHERE** caseID = 511;	Deleted client case information from client_case table where caseID is 511	Client Case with caseID 511 was deleted and no longer found in the client_case table	
22	**DELIMITER $$** **CREATE PROCEDURE** tol_client_per_department (report_start_date date, report_end_date date) **BEGIN** **SELECT** report_start_date, report_end_date, d.departmentName, count(cc.caseID) as total_clients	Created a store procedure for an inner join between service and client_case with parameters set for a given time including report_start_date and report_end_date to find total number of client cases per department in a given period. Call the store procedure for the period	The total number of client cases of 1 for ‘Public Benefits’ and 1 for ‘Financial Opportunity Center’ during ‘2021-01-01’ and ‘2021-03-31’	

	FROM department d **INNER JOIN** service s **ON** d.departmentID = s.departmentID **INNER JOIN** client_case cc ON cc.serviceID = s.serviceID **WHERE** cc.start_date **BETWEEN** report_start_date **AND** report_end_date **GROUP BY** d.departmentName; **END$$** **DELIMITER $$** **CALL** tol_client_per_department("2021-01-01", "2021-03-31");	between ‘2021-01-01’ and ‘2021-03-31’		
23	**CREATE PROCEDURE** tol_client_per_staff (report_start_date date, report_end_date date) **BEGIN** SELECT report_start_date, report_end_date, s.fName, s.lName, count(c.caseID) **FROM** staff s **INNER JOIN** client_case c **ON** s.staffID = c.staffID	Created a store procedure for an inner join between sstaff and client_case with parameters set for a given time including report_start_date and report_end_date to find total number of client cases serviced by each staff in a given period. Call the store procedure for the period between ‘2021-01-01’ and ‘2021-12-31’	The total number of client cases serviced by Kev In was 2; Siv Ian was 1; Cin Dy was 2 during ‘2021-01-01’ and ‘2021-12-31’	

	WHERE c.start_date **BETWEEN** report_start_date **AND** report_end_date **GROUP BY** s.fName, s.lName; **END$$** **DELIMITER $$;** **CALL** tol_client_per_staff ("2021-01-01", "2021-12-31");			
24	**SELECT** max(total) max_services, min(total) min_services, avg(total) avg_services **FROM** (**SELECT** count(caseID) **AS** total **FROM** Client_Case **WHERE** start_date **BETWEEN** "2021-01-01" **AND** "2021-12-31" **GROUP BY** clientID) table1;	Find the minimum, maximum and average of client cases serviced one client received for the period between ‘2021-01-01’ and ‘2021-12-31’	The minimum of client cases is 1, maximum is 2 and average is approximately 1 from ‘2021-01-01’ and ‘2021-12-31’	
25	**SELECT** d.departmentName, count(f.fundingID), sum(f.amount) **FROM** department d **INNER JOIN** funding_department fd **ON**	Inner join funding_department and funding to find the total number of new grants and the amount that eacg department received from	‘Public Benefits’ received one grant with the amount of ‘3923456.78’ from ‘2021-01-01’ and ‘021-12-31’	

	d.departmentID = fd.departmentID **INNER JOIN** funding f **ON** f.fundingID = fd.fundingID WHERE start_date BETWEEN "2021-01-01" AND "2021-12-31" GROUP BY departmentName;	‘2021-01-01’ and ‘021-12-31’		
26	**SELECT** f.funderName, fg.start_date, fg.end_date, sum(fg.amount) **FROM** funder f **INNER JOIN** funding fg ON f.funderID = fg.funderID **WHERE** DATE(fg.start_date) > "2020-01-03" **GROUP BY** f.funderName, fg.start_date, fg.end_date **ORDER BY** sum(fg.amount);	Inner Join funder and funding to find the total amount of grants after ‘2020-01-03’ funded by each funder along with start_date, end_date and sorted by the amount in the descending order	Funder Napka funded the amount of 2923456.78 starting from ‘2020-01-23’ with no end date. Funder Baker Cake funded the amount of 3923456.78 from ‘2021-01-06’ with no end date	
27	**SELECT** s.serviceType, count(cc.caseID) **FROM** Service s **INNER JOIN** Client_Case cc ON s.serviceID = cc.serviceID **WHERE** YEAR(cc.start_date) > "2005"	Inner join client_case and service to find the number of client cases serviced after year ‘2005’ by each serviceType sorted in the descending order	2 total cases for Servicetype Rent, 1 total case for each serviceType Utility, Job, Resume, Training, Informational, Education, Credit offered after year ‘2005’	

	GROUP BY s.serviceType **ORDER BY** count(cc.caseID) DESC;			
28	**SELECT** d.departmentName, count(s.staffID) **FROM** staff s **INNER JOIN** staff_department sd **ON** s.staffID = sd.staffID **INNER JOIN** department d **ON** d.departmentID = sd.departmentID **GROUP BY** d.departmentName;	Inner join staff, staff_department and department to find the total number of staff per department	Department Financial Opportunity has 6 staffs total; Public Benefits has 3 staffs total and Human Resource has 1 staff total	Passed

Conclusion:

The Financial Opportunity Center database system was designed to store data on the following tables: Staff, Department, Funding, Funder, Service, Client Case, and Client. The views and relations of the database will support the operation and collaboration for staffs under the sub-departments. The primary users for the database are: HR manager, program director, program analyst and data specialist. Using this database will allow Financial Opportunity Center to keep track of client data and client cases, facilitate communication between the departments, and improve program outcomes.

Reference:

https://lucid.app/

https://www.w3schools.com/sql/default.asp

https://www.lisc.org/our-initiatives/financial-stability/