

Step 1: Create a Family Member Table in PostgreSQL

Example:-

Create a PostgreSQL table family with at least six members, each having a minimum of seven attributes (name, age, relation, occupation, city, phone number, and email).

```
CREATE TABLE family (  
    member_id SERIAL PRIMARY KEY,  
    name TEXT,  
    age INTEGER,  
    relation TEXT,  
    occupation TEXT,  
    city TEXT,  
    phone_number TEXT,  
    email TEXT  
);
```

Insert the data for six family members.

```
INSERT INTO family (name, age, relation, occupation, city, phone_number, email)  
VALUES  
    ('Shivangi Adhikari', 20, 'Self', 'Student', 'Pune', '1234567890', 'shivangi@email.com'),  
    ('Anita Adhikari', 45, 'Mother', 'Teacher', 'Pune', '9876543210', 'anita@email.com'),  
    -- Add other family members here.
```

Step 2: Perform SQL Queries in PostgreSQL

Selection: Query to select family members based on city.

```
SELECT * FROM family WHERE city = 'Pune';
```

Data Output Messages Notifications									
	member_id [PK] integer	name text	age integer	relation text	occupation text	city text	phone_number text	email text	
1	1	Shivangi Adhikari	20	Self	Student	Pune	1234567890	shivangi@email.com	
2	2	Anita Adhikari	45	Mother	Teacher	Pune	9876543210	anita@email.com	
3	5	Priya Adhikari	16	Sister	Student	Pune	7766554433	priya@email.com	

Projection: Query to select specific fields (e.g., name and occupation).
SELECT name, occupation FROM family;

Data Output

Messages

Notifications

≡+

📄

▼

📋

▼

🗑

🗄

⬇

📈




SQL

	<div>name</div> <div>text</div> <div>🔒</div>	<div>occupation</div> <div>text</div> <div>🔒</div>
1	Shivangi Adhikari	Student
2	Anita Adhikari	Teacher
3	Amit Adhikari	Engineer
4	Rohan Sharma	Student
5	Priya Adhikari	Student
6	Ravi Adhikari	Doctor

Step 3: Export the Table to JSON Format

Export the data in JSON format from PostgreSQL:

```
COPY (SELECT json_agg(family) FROM family) TO 'path/to/your/file.json';  
// path of ur folder
```

	RedNotebook	11-10-2024 09:53	File folder	
	WebStorm 2024.2.3	22-10-2024 22:05	File folder	
	mynosqlthing	27-10-2024 15:52	JetBrains WebStorm	1 KB

Step 4: Import JSON into MongoDB

Import the JSON document into MongoDB using MongoDB Compass or the shell.

Step 5: Perform MongoDB Operations

****Insert a new document:**

Example

```
db.family.insertOne({
  "name": "Priya Adhikari",
  "age": 16,
  "relation": "Sister",
  "occupation": "Student",
  "city": "Pune",
  "phone_number": "5566778899",
  "email": "priya@email.com"
});
```

family > family_db > family

Documents 7 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query](#)

ADD DATA EXPORT DATA UPDATE DELETE

```
name : "Anita Adhikari"
age : 45
relation : "Mother"
occupation : "Teacher"
city : "Pune"
phone_number : "9876543210"
email : "anita@email.com"
```

```
_id: ObjectId('671e17ed41ee6b3038f9efbe')
member_id : 5
name : "Priya Adhikari"
age : 16
relation : "Sister"
occupation : "Student"
city : "Pune"
phone_number : "7766554433"
email : "priya@email.com"
```

```
_id: ObjectId('671e182941ee6b3038f9efc1')
name : "Priya Adhikari"
age : 16
relation : "Sister"
occupation : "Student"
city : "Pune"
phone_number : "5566778899"
email : "priya@email.com"
```

****Search for documents:**
db.family.find({ "city": "Pune" });

family > family_db > family

Documents 7 Aggregations Schema Indexes 1 Validation

{ "city": "Pune" } [Generate query](#) [Explain](#) [Reset](#) [Find](#)

[+ ADD DATA](#) [EXPORT DATA](#) [UPDATE](#) [DELETE](#)

25 1 - 4 of 4

```
_id: ObjectId('671e17ed41ee6b3038f9efba')
member_id: 1
name: "Shivangi Adhikari"
age: 20
relation: "Self"
occupation: "Student"
city: "Pune"
phone_number: "1234567890"
email: "shivangi@email.com"
```

```
_id: ObjectId('671e17ed41ee6b3038f9efbb')
member_id: 2
name: "Anita Adhikari"
age: 45
relation: "Mother"
occupation: "Teacher"
city: "Pune"
phone_number: "9876543210"
email: "anita@email.com"
```

```
_id: ObjectId('671e17ed41ee6b3038f9efbe')
member_id: 5
name: "Priya Adhikari"
age: 16
relation: "Sister"
occupation: "Student"
```

****Update a document: example**
db.family.updateOne(
 { "name": "Amit Adhikari" },
 { \$set: { "occupation": "Engineer" } }
);

1	_id: ObjectId('671e17ed41ee6b3038f9efbc')	ObjectId
2	member_id: 3	Int32
3	name: "Amit Adhikari"	String
4	age: 50	Int32
5	relation: "Father"	String
6	occupation: "Scientist"	String
7	city: "Mumbai"	String
8	phone_number: "5566778899"	String
9	email: "amit@email.com"	String

Document modified.

[CANCEL](#) [UPDATE](#)

**** Delete a document:example**

```
db.family.deleteOne({ "name": "Rohan Sharma" });
```

```
_id: ObjectId('671e182941ee6b3038f9efc1')
name: "Priya Adhikari"
age: 16
relation: "Sister"
occupation: "Student"
city: "Pune"
phone_number: "5566778899"
email: "priya@email.com"
```

Document flagged for deletion.

CANCEL DELETE

Step 6: Implement Aggregation Pipeline in MongoDB

Example

Create an aggregation pipeline with at least three stages:

```
db.family.aggregate([
  { $match: { "city": "Pune" } },
  { $group: { _id: "$occupation", count: { $sum: 1 } } },
  { $project: { _id: 0, occupation: "$_id", count: 1 } }
]);
```

family > family_db > family

Documents 6 Aggregations Schema Indexes 1 Validation

Smatch Sgroup Sproject Generate aggregation + ? Explain Export Run Options ▶

Untitled - modified SAVE + CREATE NEW EXPORT TO LANGUAGE PREVIEW STAGES TEXT WIZARD

```
1 [
2   { "$match": { "city": "Pune" } },
3   { "$group": { "_id": "$occupation", "count": { "$sum": 1 } } },
4   { "$project": { "_id": 0, "occupation": "$_id", "count": 1 } }
5 ]
6
```

PIPELINE OUTPUT

Sample of 2 documents

count : 2
occupation : "Student"

count : 1
occupation : "Teacher"