Let's create some relational algebra queries for a hospital management system. I'll define a simplified schema first.

Simplified Schema:

- **Patient:** (PatientID, Name, Address, Phone, DOB)
- **Doctor:** (DoctorID, Name, Specialization)
- **Appointment:** (AppointmentID, PatientID, DoctorID, Date, Time, Diagnosis)
- **Department:** (DepartmentID, DepartmentName, Location)
- **DoctorDepartment:** (DoctorID, DepartmentID) *(To link doctors to departments)*

Example Queries and Relational Algebra:

1. **Find the names and addresses of all patients:**

   Code snippet

   ```
   π Name, Address (Patient)
   ```

2. **Find the names of all doctors specializing in Cardiology:**

   Code snippet

   ```
   π Name (σ Specialization = 'Cardiology' (Doctor))
   ```

3. **Find the names of all patients who have an appointment with Dr. Smith:**

   Code snippet

   ```
   π Name (Patient ⋈ PatientID (σ DoctorID = (SELECT DoctorID FROM Doctor
   WHERE Name = 'Dr. Smith') (Appointment)))
   ```

   (Note: The subquery finds Dr. Smith's ID)

4. **Find the date and time of all appointments for patient 'John Doe':**

   Code snippet

   ```
   π Date, Time (Appointment ⋈ PatientID (σ Name = 'John Doe' (Patient)))
   ```

5. **Find the names of all doctors in the Cardiology department:**

   Code snippet

   ```
   π Name (Doctor ⋈ DoctorID (DoctorDepartment ⋈ DepartmentID (σ
   DepartmentName = 'Cardiology' (Department))))
   ```

   (This joins Doctor, DoctorDepartment, and Department to link doctors to departments)

6. **Find the names of all patients who have an appointment on '2024-03-15':**

Code snippet

```
π Name (Patient ⋈ PatientID (σ Date = '2024-03-15' (Appointment)))
```

7. **Find the names of all doctors who have seen at least one patient:**

   Code snippet

   ```
   π Name (Doctor ⋈ DoctorID Appointment)
   ```

8. **Find the names of all patients who have *not* had any appointments:**

   Code snippet

   ```
   π Name (Patient) - π Name (Patient ⋈ PatientID Appointment)
   ```

9. **Find the names of doctors and the number of appointments they have:** This one is a bit more complex and usually requires aggregation, which is not strictly part of basic relational algebra. However, some extended versions have aggregation operators. Here's a conceptual representation:

```
10._G DoctorID; count(AppointmentID) (Doctor ⋈ DoctorID Appointment)
```

   (`_G` represents grouping, and `count` is an aggregate function. This is a simplified representation of how aggregation might be incorporated). In SQL, this would be done with `GROUP BY` and `COUNT`.

11. **Find the name of the department located in 'New York':**

    Code snippet

    ```
    π DepartmentName (σ Location = 'New York' (Department))
    ```