Integrator:

I'm asked to integrate with a PostgreSQL server.

First step is to pull PostgreSQL from docker, then setup a docker container with a PostgreSQL image:

docker pull postgres
winpty docker run -it --rm postgres bash

Then update and install postgresql-client
apt-get update
apt-get install -y postgresql-client

Then use the connection string provided by exposer.
PGPASSWORD=<password> psql -h <server_name> -U <username> -p <port> -d <dbname> psql

I am now connected to the database, so let's list all tables.

```
railway=> \dt
          List of relations
 Schema |    Name    | Type  |  Owner
--------+------------+-------+----------
 public | department | table | postgres
 public | employee   | table | postgres
 public | project    | table | postgres
(3 rows)
```

I am told that three users are set up for the database with different rights. I am currently logged in as admin

```
railway=> \du
                                   List of roles
   Role name    |                         Attributes
----------------+------------------------------------------------------------
 admin          |
 emp1           |
 postgres       | Superuser, Create role, Create DB, Replication, Bypass RLS
 read_only_dept |
```

I am told that the admin user has full access to all tables. There's a readonly user that can only read from the department table, and an emp1 user that can only access employees.

Let’s try some statements from the admin user:

```
railway=> select * from employee;
 employee_id | employee_name |        email        | department_id
-------------+---------------+---------------------+---------------
           1 | Alice         | alice@company.com   |             1
           2 | Bob           | bob@company.com     |             1
           3 | Charlie       | charlie@company.com |             2
           4 | Daisy         | daisy@company.com   |             3
(4 rows)
```

```
railway=> insert into employee(employee_id, employee_name, email, department_id)
values(5, 'John Doe', 'johndoe@company.com', 3);
INSERT 0 1
railway=> select * from employee;
 employee_id | employee_name |        email        | department_id
-------------+---------------+---------------------+---------------
           1 | Alice         | alice@company.com   |             1
           2 | Bob           | bob@company.com     |             1
           3 | Charlie       | charlie@company.com |             2
           4 | Daisy         | daisy@company.com   |             3
           5 | John Doe      | johndoe@company.com |             3
(5 rows)
```

```
railway=> select * from employee;
 employee_id | employee_name |        email        | department_id
-------------+---------------+---------------------+---------------
           1 | Alice         | alice@company.com   |             1
           2 | Bob           | bob@company.com     |             1
           3 | Charlie       | charlie@company.com |             2
           4 | Daisy         | daisy@company.com   |             3
           5 | John Doe      | john@company.com    |             3
(5 rows)
```

```
railway=> delete from employee where employee_id = 5;
DELETE 1
railway=> select * from employee;
 employee_id | employee_name |        email        | department_id
-------------+---------------+---------------------+---------------
           1 | Alice         | alice@company.com   |             1
           2 | Bob           | bob@company.com     |             1
           3 | Charlie       | charlie@company.com |             2
           4 | Daisy         | daisy@company.com   |             3
(4 rows)
```

read_only_dept user:

```
railway=> select current_user;
  current_user
---------------
 read_only_dept
(1 row)
```

Let’s try inserting into another table:

```
railway=> insert into employee(employee_id, employee_name, email, department_id)
values(5, 'John Doe', 'johndoe@company.com', 3);
ERROR:  permission denied for table employee
```

We successfully get denied.

Delete:

```
railway=> delete from employee where employee_id = 4;
ERROR:  permission denied for table employee
```

Update:

```
railway=> update employee set email = '1@1.com' where employee_id = 4;
ERROR:  permission denied for table employee
```

Department table:

```
railway=> select * from department;
 department_id | department_name
---------------+-----------------
             1 | Human Resources
             2 | IT
             3 | Marketing
(3 rows)

railway=> insert into department(department_id, department_name) VALUES (4, 'Finance');
ERROR:  permission denied for table department
railway=> update department set department_name = 'Finance' where department_id = 1;
ERROR:  permission denied for table department
railway=> delete from department where department_id = 3;
ERROR:  permission denied for table department
```

According to the documentation, this is correct.

Last is emp1 user with employee-level access:

```
railway=> select current_user;
 current_user
--------------
 emp1
(1 row)
```

This user can only access employees/projects connected to department 1.

```
railway=> select * from employee;
 employee_id | employee_name |       email       | department_id
-------------+---------------+-------------------+---------------
           1 | Alice         | alice@company.com |             1
           2 | Bob           | bob@company.com   |             1
(2 rows)
```

```
railway=> select * from project;
 project_id |    project_name    | start_date |  end_date  | department_id
------------+--------------------+------------+------------+---------------
          1 | Onboarding Process | 2023-01-01 | 2023-06-30 |             1
(1 row)
```

Lets try inserting an employee with another department_id than 1.

```
railway=> insert into employee(employee_id, employee_name, email, department_id)
values(5, 'John Doe', 'johndoe@company.com', 2);
ERROR:  new row violates row-level security policy for table "employee"
```

We successfully get denied.

Lets try inserting an employee with department_id 1 instead.

```
railway=> insert into employee(employee_id, employee_name, email, department_id)
values(5, 'John Doe', 'johndoe@company.com', 1);
INSERT 0 1
railway=> select * from employee;
 employee_id | employee_name |        email        | department_id
-------------+---------------+---------------------+---------------
           1 | Alice         | alice@company.com   |             1
           2 | Bob           | bob@company.com     |             1
           5 | John Doe      | johndoe@company.com |             1
(3 rows)
```

It works.

Lets update John Doe's email:

```
railway=> update employee set email = '1@1.com' where employee_id = 5;
UPDATE 1
railway=> select * from employee;
 employee_id | employee_name |       email       | department_id
-------------+---------------+-------------------+---------------
           1 | Alice         | alice@company.com |             1
           2 | Bob           | bob@company.com   |             1
           5 | John Doe      | 1@1.com           |             1
(3 rows)
```

Let's try updating an employee we know exists, but is not connected to department 1:
From previous queries, we know there's an employee with id = 3.

```
railway=> update employee set email = 'company@company.com' where employee_id = 3;
UPDATE 0
```

Since we get UPDATE 0, no rows were updated, and that's good.

Let's try to delete the same user we tried to update.

```
railway=> delete from employee where employee_id = 3;
DELETE 0
```

Lastly we can delete am employee connected to department 1.

```
railway=> select * from employee;
 employee_id | employee_name |       email       | department_id
-------------+---------------+-------------------+---------------
           1 | Alice         | alice@company.com |             1
           2 | Bob           | bob@company.com   |             1
           5 | John Doe      | 1@1.com           |             1
(3 rows)

railway=> delete from employee where employee_id = 5;
DELETE 1
```