Privileges

When an object is created, it is assigned an owner. The owner is normally the role that executed the creation statement. For most kinds of objects, the initial state is that only the owner (or a superuser) can do anything with the object. To allow other roles to use it, *privileges* must be granted. There are several different kinds of privilege: `SELECT`, `INSERT`, `UPDATE`, `DELETE`, `TRUNCATE`, `REFERENCES`, `TRIGGER`, `CREATE`, `CONNECT`, `TEMPORARY`, `EXECUTE`, and `USAGE`. For more information on the different types of privileges supported by PostgreSQL, see the **GRANT** reference page.

To assign privileges, the `GRANT` command is used. So, if `joe` is an existing role, and `accounts` is an existing table, the privilege to update the table can be granted with:

```
GRANT UPDATE ON accounts TO joe;
```

The special name `PUBLIC` can be used to grant a privilege to every role on the system. Writing `ALL` in place of a specific privilege specifies that all privileges that apply to the object will be granted.

To revoke a privilege, use the fittingly named **REVOKE** command:

```
REVOKE ALL ON accounts FROM PUBLIC;
```

The special privileges of an object's owner (i.e., the right to modify or destroy the object) are always implicit in being the owner, and cannot be granted or revoked. But the owner can choose to revoke his own ordinary privileges, for example to make a table read-only for himself as well as others.

An object can be assigned to a new owner with an `ALTER` command of the appropriate kind for the object. Superusers can always do this; ordinary roles can only do it if they are both the current owner of the object (or a member of the owning role) and a member of the new owning role.

Obtenido de: https://www.postgresql.org/docs/8.4/static/privileges.html
Consultado: 27 de enero de 2017