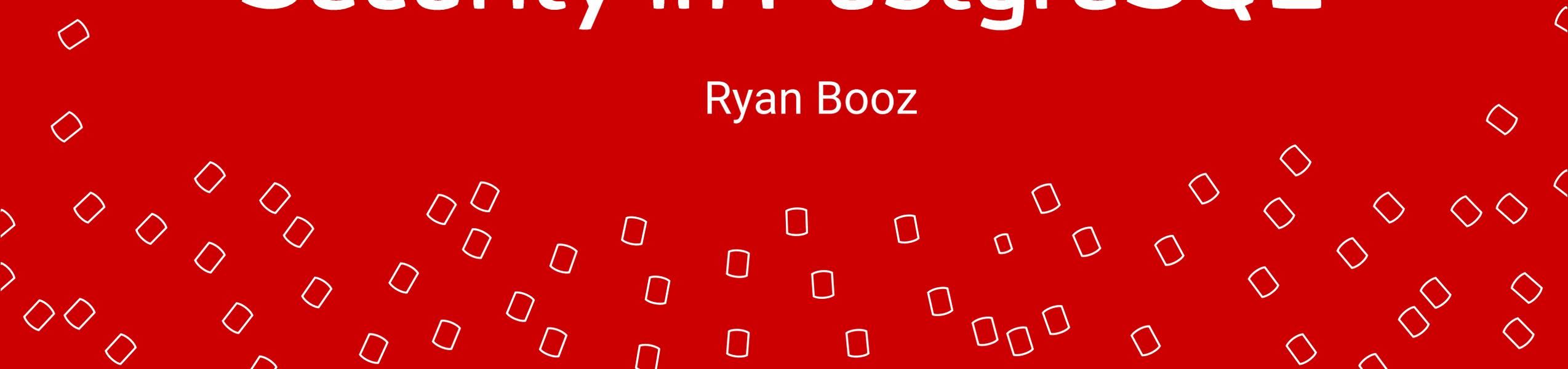


For Your Eyes Only: Roles, Privileges, and Security in PostgreSQL

Ryan Booz



Ryan Booz

**PostgreSQL & DevOps
Advocate**



[@ryanbooz](https://twitter.com/@ryanbooz)



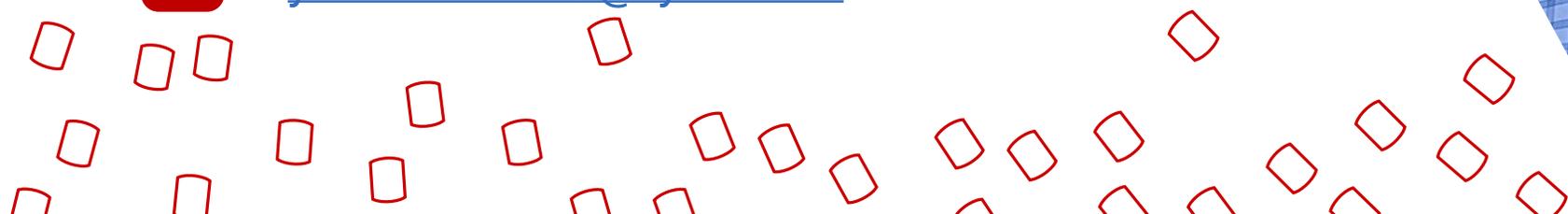
[/in/ryanbooz](https://www.linkedin.com/in/ryanbooz)



www.softwareandbooz.com



youtube.com/@ryanbooz





redgate

Simple Talk



A technical journal and community hub

✓ Blogs, videos, events, and more

✓ Contributions by experts

✓ Unbiased, multi-database content

Join the community at Simple-Talk.com

github.com/ryanbooz/presentations

Agenda

- 01 The Building Blocks
- 02 Roles
- 03 Special Roles
- 04 Privileges
- 05 Inheritance
- 06 Object Ownership
- 07 Predefined Roles

Disclaimer(s)

We won't cover everything

Content is applicable to
currently supported versions
of PostgreSQL (12+)

(and most of it for 9.6+)

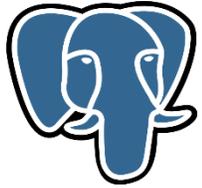


01/07

The Building Blocks



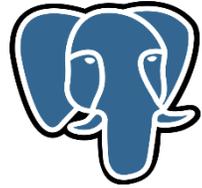
Server/Host (Firewall, Ports)



Cluster

Port: 5432

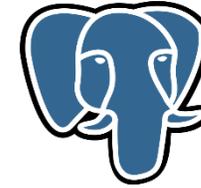
pg_hba.conf



Cluster

Port: 5433

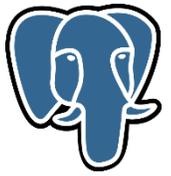
pg_hba.conf



Cluster

Port: 5434

pg_hba.conf



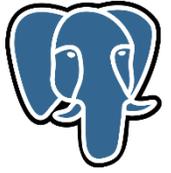
Cluster

ROLES



Databases





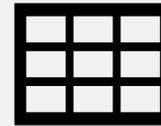
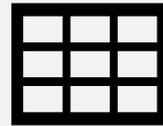
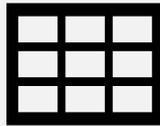
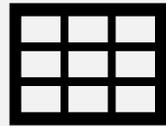
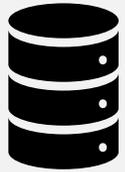
Cluster



ROLE



Database



pg_hba.conf

- First layer of authentication
- Similar to a firewall ruleset for PostgreSQL
- Cloud vendors largely manage this for you

**Which hosts & roles, can connect to what databases,
using what authentication method?**

**Avoid using 'TRUST'
method at all costs!***

***(can be useful for local development machines... but still...)**

Use scram-sha-256 for
password authentication

02/07 Roles

Roles

- Own databases, schemas, and objects
 - Tables, Functions, Views, Etc.
- Have cluster-level privileges (attributes)
- Granted privileges to databases, schemas, and objects
- Can possibly grant privileges to other roles



Users and Groups

- Semantically the same as roles
- `CREATE USER`
 - `CREATE USER LOGIN`
 - `CREATE USER NOLOGIN`
- PostgreSQL 8.2+ `CREATE (USER | GROUP)` is an alias

```
CREATE USER user1 WITH PASSWORD 'abc123' INHERIT;
```

```
CREATE GROUP group1 WITH INHERIT;
```

```
CREATE ROLE user1 WITH LOGIN PASSWORD 'abc123' INHERIT;
```

Role Attributes

- Predefined settings that can be enabled/disabled for a given role
- Essentially cluster-level (non-database) privileges
- Map to columns in `pg_catalog.pg_roles`

PostgreSQL 15 Attributes

LOGIN

PASSWORD

SUPERUSER

INHERIT

CREATEROLE

BYPASSRLS

CREATEDB

CONNECTION LIMIT

REPLICATION LOGIN

Unless otherwise set, new roles can INHERIT privileges from other roles and have unlimited connections

Role Specific Session Settings

- “ALTER ROLE user1 SET jit TO off;
ALTER ROLE user1 RESET jit;
ALTER ROLE user1 SET jit TO on;
ALTER ROLE user1 RESET jit;”
- ALTER ROLE user1 SET jit TO off;
ALTER ROLE user1 RESET jit;
ALTER ROLE user1 SET jit TO on;
ALTER ROLE user1 RESET jit;

```
ALTER ROLE user1 SET jit TO off;
```

```
ALTER ROLE user1 RESET jit;
```

03/07
Special Roles

PostgreSQL Superuser



[This Photo](#) by Unknown Author is licensed under [CC BY-NC-ND](#)

PostgreSQL Superuser



PostgreSQL Superuser

- 🧑 is created by default when the cluster is initialized
- Typically named `postgres` because the system process user initiates a `initdb`
- Bypasses all security checks except `LOGIN`
- Full privilege to do "anything"
- Treat superuser with care (like `root` on Linux)

**Most cloud providers do not
provide superuser access**

Superuser-like



Superuser-like

- Create a role with the right level of control
- Recommend adding `CREATEROLE` and `CREATEDB`
- `CREATE ROLE superuser WITH LOGIN SUPERUSER CREATEDB CREATEROLE;`
- `GRANT superuser TO user;`

04/07
Privileges

Privileges

- The set of access rights to databases, schemas, and objects
- Can be granted (`GRANT`) or revoked (`REVOKE`) by a role with authority
- Explicit `GRANT` or `REVOKE` only impacts existing objects

PostgreSQL 15+ Privileges

SELECT

INSERT

UPDATE

DELETE

TRUNCATE

REFERENCES

TRIGGER

CREATE

CONNECT

TEMPORARY

EXECUTE

USAGE

SET

ALTER SYSTEM

Security Best Practice for PUBLIC

- Revoke all privileges on the public schema from the `PUBLIC` role
- Revoke all database privileges from the `PUBLIC` role (maybe)

```
REVOKE ALL ON SCHEMA public FROM PUBLIC;  
REVOKE ALL ON DATABASE db_name FROM PUBLIC;
```

Granting Privileges

```
-- grant the ability to create a schema
GRANT CREATE ON DATABASE app_db TO admin1;

-- see and create objects in schema
GRANT USAGE,CREATE IN SCHEMA demo_app TO dev1;

-- allow some roles only some privileges
GRANT SELECT,INSERT,UPDATE
ON ALL TABLES IN SCHEMA demo_app TO jr_dev;
```

Granting Privileges

- Remember, explicit grants only effect existing database objects!

```
-- This will only grant to existing objects  
GRANT ALL TO ALL TABLES IN SCHEMA public TO dev1;
```

More Detail on GRANT and REVOKE

What the privileges mean:

<https://www.postgresql.org/docs/current/ddl-priv.html>

How to GRANT privileges:

<https://www.postgresql.org/docs/current/sql-grant.html>

How to REVOKE privileges:

<https://www.postgresql.org/docs/current/sql-revoke.html>

05/07

Inheritance

Privilege Inheritance

- Roles can be granted membership into another role
- If a role has `INHERIT` set, they automatically have usage of privileges from member roles
- The preferred method for managing group privileges

Granting Privileges



```
CREATE ROLE sr_dev WITH LOGIN password='abc' INHERIT;  
CREATE ROLE rptusr WITH LOGIN password='123' INHERIT;  
CREATE ROLE admin WITH NOLOGIN NOINHERIT;  
CREATE ROLE ropriv WITH NOLOGIN NOINHERIT;
```



```
GRANT INSERT,UPDATE,DELETE ON ALL TABLES  
  IN SCHEMA app TO admin;  
GRANT SELECT ON ALL TABLES IN SCHEMA app TO ropriv;
```

```
GRANT admin,ropriv TO sr_dev;  
GRANT ropriv TO rptusr;
```

Table access on 'app' schema



ropriv

rptusr



SELECT

sr_dev



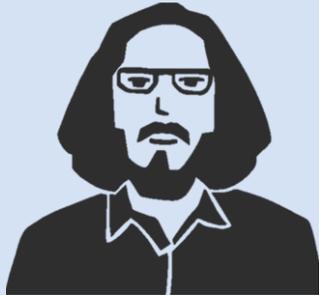
SELECT

Table access on 'app' schema



ropriv

rptusr



SELECT



admin

sr_dev



**SELECT, INSERT,
UPDATE, DELETE**

06/07

Object Ownership

Object Ownership

- Object creator = owner
- Owner is a "superuser" of the objects they own
- Initial object access = **Principle of Least Privilege**
 - Unless specifically granted ahead of time, objects are owned and "accessible" by the creator/superuser only
- Roles can specify default privileges to `GRANT` for each object type that they create



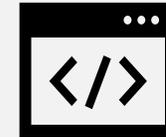
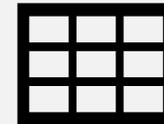
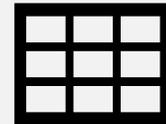
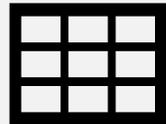
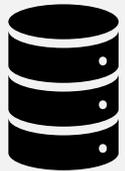
Cluster

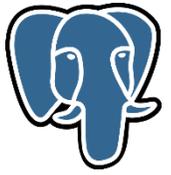


ROLE



Database





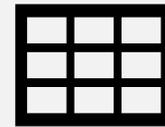
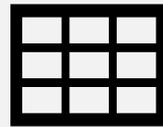
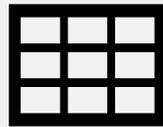
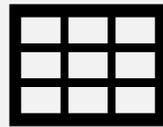
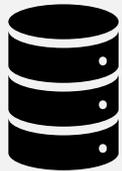
Cluster



ROLE



Database



Default Privileges

```
ALTER DEFAULT PRIVILEGES  
GRANT SELECT ON TABLES TO public;
```

```
role_demo=> \ddp
```

```
          Default access privileges
```

Owner	Schema	Type	Access privileges
-----+-----+-----+-----			
postgres		table	=r/postgres +
			postgres=arwdDxt/postgres

Providing Object Access

Option 1: (owner)

Explicitly GRANT access
after object creation

Option 2: (owner)

ALTER DEFAULT
PRIVILEGES

Option 3:

SET ROLE to **app role**
before creation with
correct **default privileges**

Option 4: (PG14+)

Use pg_read_all_data
or pg_write_all_data
predefined roles

Object Ownership Security

- `CREATE OR REPLACE` doesn't change ownership
- Security issue with users that have create permissions (particularly the `public` schema)
- PostgreSQL 15 removes default create permissions from PUBLIC on the public schema

DEMO

07/07

Predefined Roles

Predefined Roles

- Cluster-level roles that can be granted
- Work starting in PostgreSQL 14+ to simplify privilege management
- `pg_read_all_data` (for example)
 - If a role that has `CONNECT` to a database, they can `SELECT` from all tables

Table 22.1. Predefined Roles

Role	Allowed Access
pg_read_all_data	Read all data (tables, views, sequences), as if having <code>SELECT</code> rights on those objects, and <code>USAGE</code> rights on all schemas, even without having it explicitly. This role does not have the role attribute <code>BYPASSRLS</code> set. If RLS is being used, an administrator may wish to set <code>BYPASSRLS</code> on roles which this role is <code>GRANTED</code> to.
pg_write_all_data	Write all data (tables, views, sequences), as if having <code>INSERT</code> , <code>UPDATE</code> , and <code>DELETE</code> rights on those objects, and <code>USAGE</code> rights on all schemas, even without having it explicitly. This role does not have the role attribute <code>BYPASSRLS</code> set. If RLS is being used, an administrator may wish to set <code>BYPASSRLS</code> on roles which this role is <code>GRANTED</code> to.
pg_read_all_settings	Read all configuration variables, even those normally visible only to superusers.
pg_read_all_stats	Read all <code>pg_stat_*</code> views and use various statistics related extensions, even those normally visible only to superusers.
pg_stat_scan_tables	Execute monitoring functions that may take <code>ACCESS SHARE</code> locks on tables, potentially for a long time.
pg_monitor	Read/execute various monitoring views and functions. This role is a member of <code>pg_read_all_settings</code> , <code>pg_read_all_stats</code> and <code>pg_stat_scan_tables</code> .
pg_database_owner	None. Membership consists, implicitly, of the current database owner.
pg_signal_backend	Signal another backend to cancel a query or terminate its session.
pg_read_server_files	Allow reading files from any location the database can access on the server with <code>COPY</code> and other file-access functions.
pg_write_server_files	Allow writing to files in any location the database can access on the server with <code>COPY</code> and other file-access functions.
pg_execute_server_program	Allow executing programs on the database server as the user the database runs as with <code>COPY</code> and other functions which allow executing a server-side program.
pg_checkpoint	Allow executing the <code>CHECKPOINT</code> command.
pg_use_reserved_connections	Allow use of connection slots reserved via <code>reserved_connections</code> .
pg_create_subscription	Allow users with <code>CREATE</code> permission on the database to issue <code>CREATE SUBSCRIPTION</code> .

<https://www.postgresql.org/docs/current/predefined-roles.html>

What Questions do you have?



redgate

Simple Talk



A technical journal and community hub

✓ Blogs, videos, events, and more

✓ Contributions by experts

✓ Unbiased, multi-database content

Join the community at Simple-Talk.com

 THANK YOU! 

github.com/ryanbooz/presentations