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

Ryan Booz

Scale20x
March 2023

redgate

redgate | About me

Ryan Booz

PostgreSQL & DevOps Advocate

@ryanbooz

/in/ryanbooz

www.softwareandbooz.com

github.com/ryanbooz/presentations

redgate

👫 +[👧 , 👧 , 👦 , 👧 , 👦 , 👧] = ❤️

👨‍👩‍👧‍👦 , 🎵 , 🐝 , ☕ = 💯

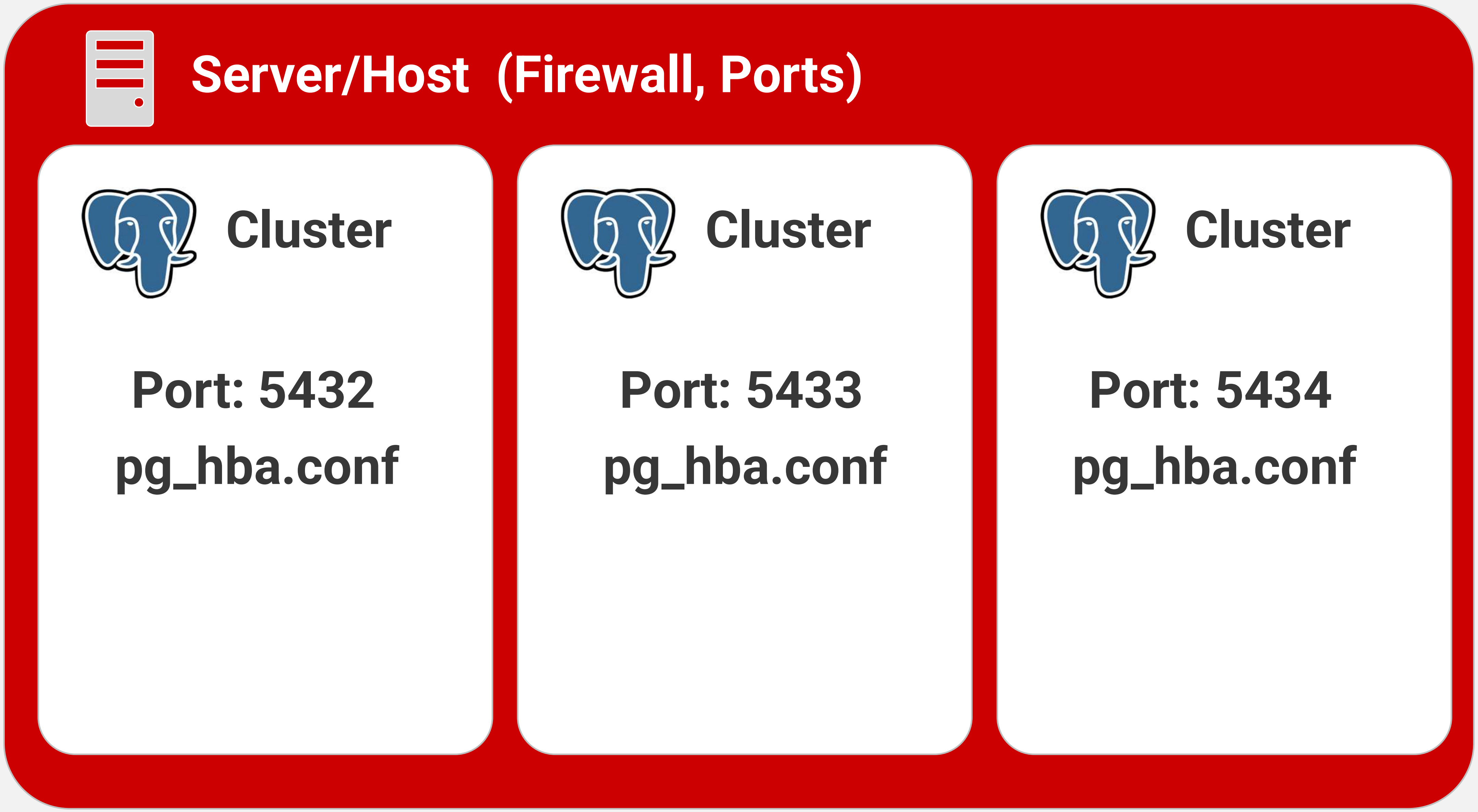
Agenda

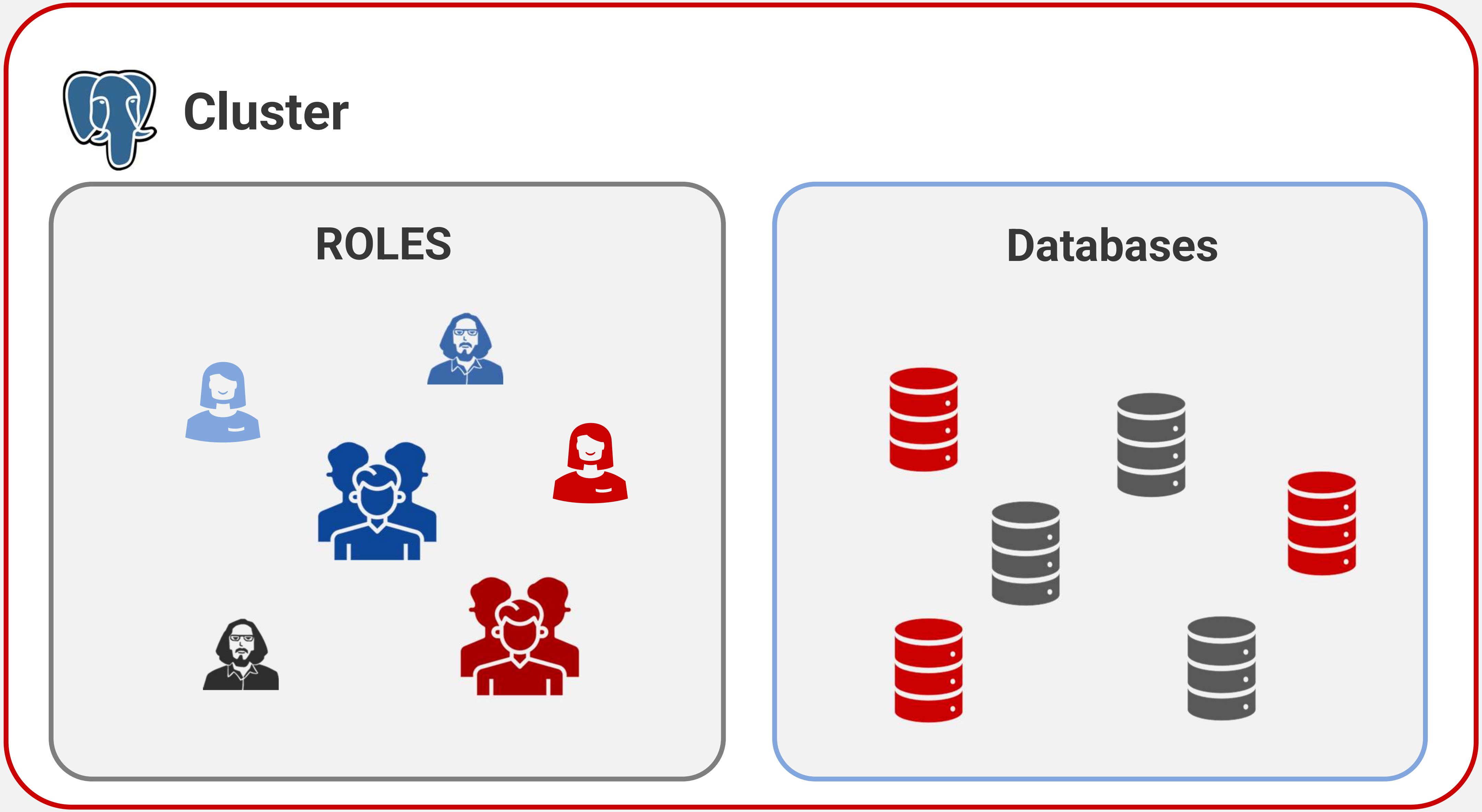
redgate

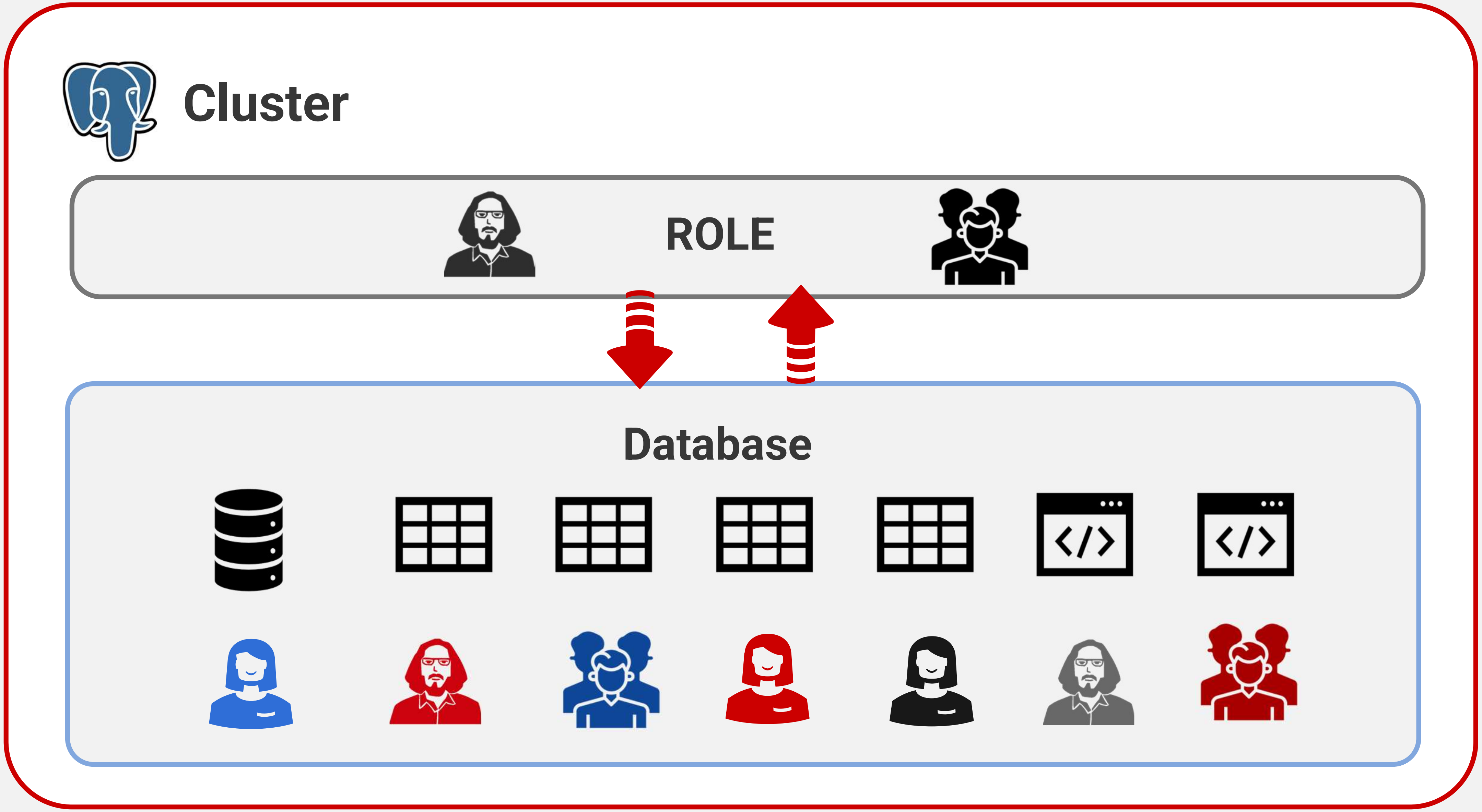
01/07

The Building Blocks

redgate


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

Authentication (AuthN)

Is the role who they say they are?

redgate

pg_hba.conf

- First layer of authentication
- Similar to a firewall ruleset for PostgreSQL

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

redgate

```
# Allow any user on the local system to connect to any database with
# any database user name using Unix-domain sockets (the default for local
# connections).
#
# TYPE  DATABASE        USER            ADDRESS                 METHOD
local   all             all                                     trust

# The same using local loopback TCP/IP connections.
#
# TYPE  DATABASE        USER            ADDRESS                 METHOD
host    all             all             127.0.0.1/32            trust

# Allow any user from host 192.168.12.10 to connect to database
# "postgres" if the user's password is correctly supplied.
#
# TYPE  DATABASE        USER            ADDRESS                 METHOD
host    postgres        all             192.168.12.10/32        scram-sha-256
```

https://www.postgresql.org/docs/current/auth-pg-hba-conf.html

Avoid using 'TRUST' at (almost) all costs!

Use scram-sha-256 for password authentication

02/07

Roles

redgate

Roles

- Own database objects
  - Tables, Functions, Etc.
- Have cluster-level privileges (attributes)
- Granted privileges to database objects
- Can grant privileges to other roles

Users and Groups

- Semantically the same as roles
- By Convention:
  - User = `LOGIN`
  - Group = `NOLOGIN`
- PostgreSQL 8.2+ `CREATE (USER|GROUP)` is an alias

redgate

```
CREATE USER us        WI        ASSWOR      bc123' INHERIT;
```

```
CREATE         P group          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

SUPERUSER

CREATEROLE

CREATEDB

REPLICATION LOGIN

PASSWORD

INHERIT

BYPASSRLS

CONNECTION LIMIT

redgate

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

redgate

03/07

Special Roles

redgate

PostgreSQL Superuser

PostgreSQL Superuser

PostgreSQL Superuser

- 🦸 is created by default when the cluster is initialized
- Typically named `postgres` after the account that owns the server processes
- Bypasses all security checks except `LOGIN`
- Full privilege to do "anything"
- Treat superuser with care (like `root` on Linux)

Most cloud provides do not provide superuser access*

*And Magnus said you shouldn't give yourself superuser anyway!

redgate

Superuser-like

Superuser-like

- Create a role with the right level of control
- Recommend adding `CREATEROLE` and `CREATEDB`
- Allows user management and database ownership
- May still limit some actions (e.g. installing extensions limited to superuser)

redgate

PUBLIC Role

- All roles are granted implicit membership to `PUBLIC`
- The public role cannot be deleted
- Granted `CONNECT`, `USAGE`, `TEMPORARY`, and `EXECUTE` by default
- <=PG14: `CREATE` on the public schema by default
- >=PG15: No `CREATE` on public schema by default

redgate

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;
```

04/07

Privileges

Authorization (AuthZ)

As an authenticated user, what am I allowed to do within the system?

redgate

Principle of Least Privileges (PoLP)

- Roles are provided the least amount of access by default
- Every role must be given explicit access
- Only owners of objects (or superusers) can alter access and privileges
- Only owners of objects (or superusers) can alter objects

Privileges

- The set of access rights to databases and database 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

redgate

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 app TO dev1;

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

redgate

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;
```

'Since the role identity determines the set of privileges available to a connected client, it is important to carefully configure privileges when setting up a multiuser environment.'

- PostgreSQL Documentation

More Detail on GRANT and REVOKE

What the permissions 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

redgate

05/07

Inheritance

redgate

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

redgate

Granting Privileges

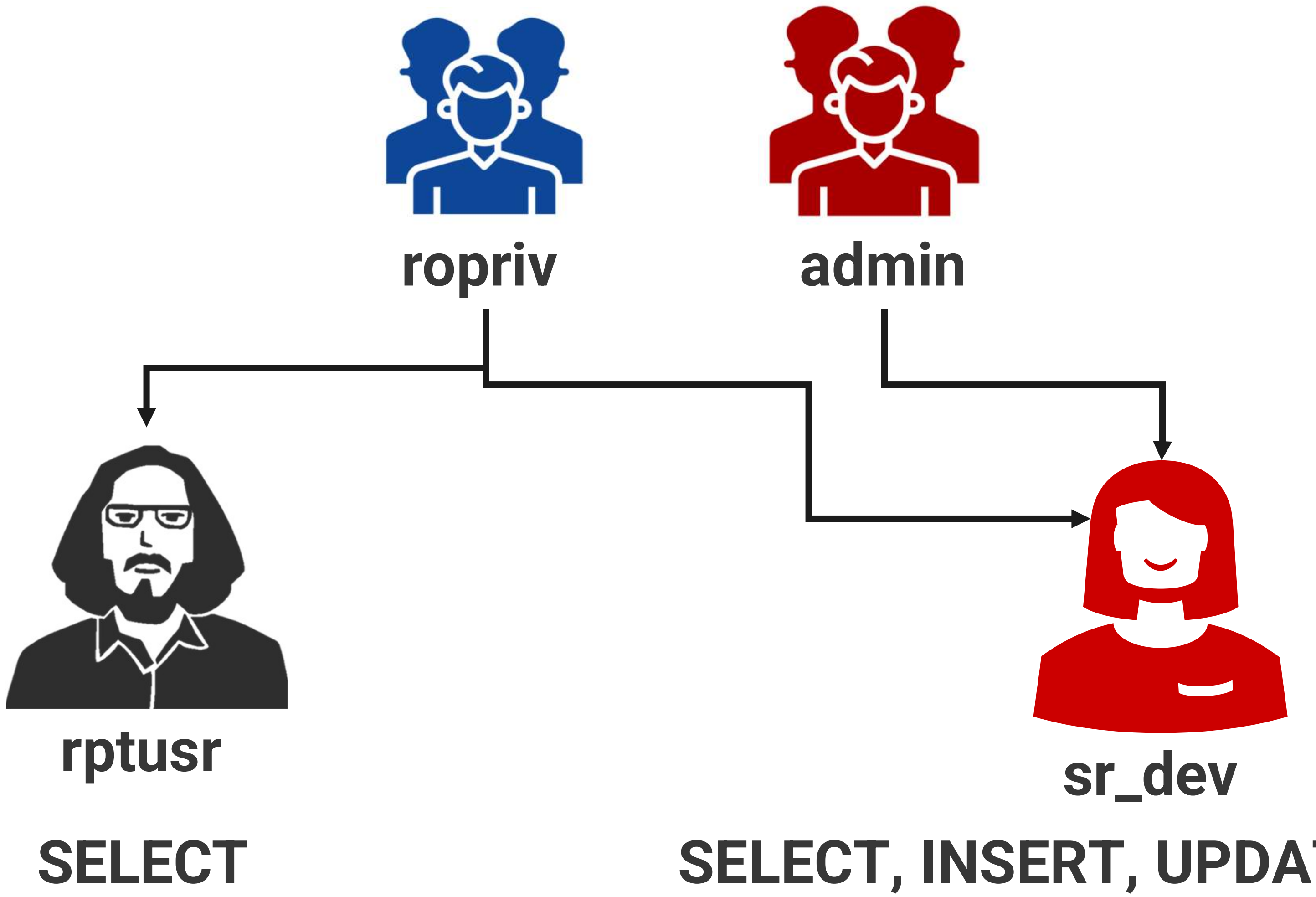
```
CREATE ROLE admin WITH NOLOGIN NOINHERIT;
GRANT INSERT,UPDATE,DELETE ON ALL TABLES
IN SCHEMA app TO admin;

CREATE ROLE ropriv WITH NOLOGIN NOINHERIT;
GRANT SELECT ON ALL TABLES IN SCHEMA app TO ropriv;

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

redgate

Table access on 'app' schema



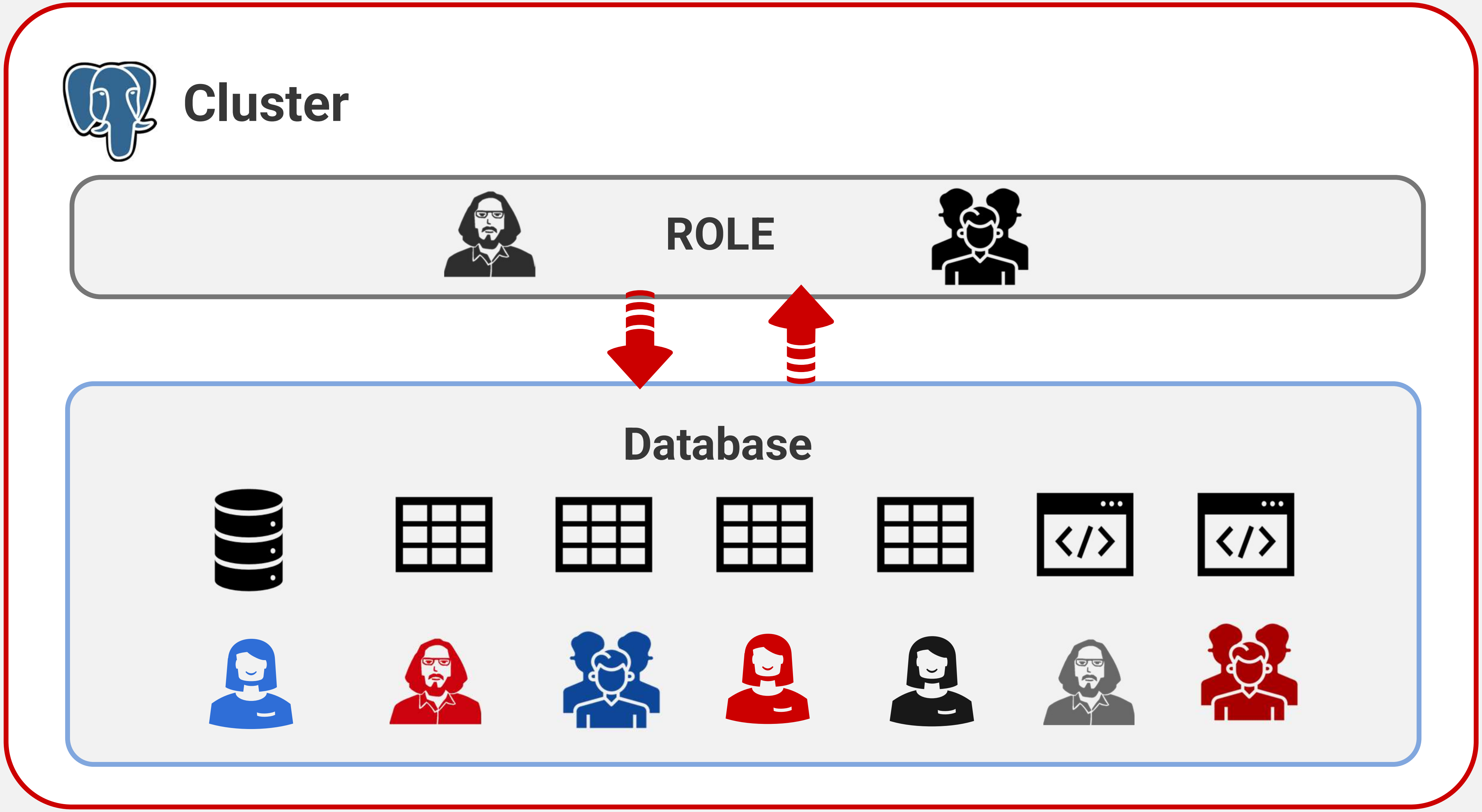
06/07

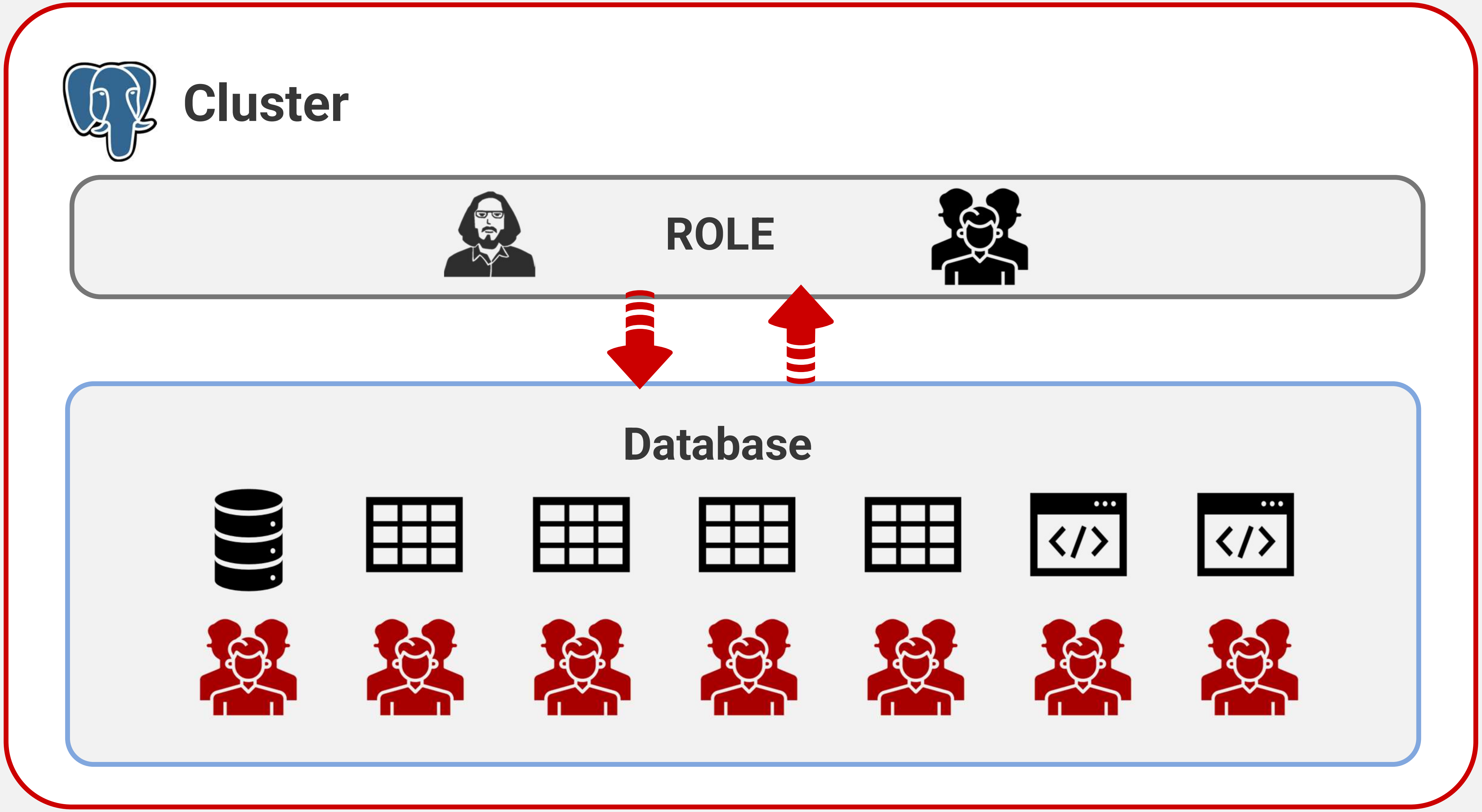
Object Ownership

redgate

Object Ownership

- Object creator = owner
- Initial object access = **Principle of Least Privilege**
  - Unless specifically granted ahead of time, objects are owned and "accessible" by the creator 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;


scale20x=> \ddp
                Default access privileges
  Owner    | Schema | Type  |     Access privileges
-----------+--------+-------+-------------------------
 postgres  |        | table | =r/postgres            +
           |        |       | postgres=arwdDxt/postgres
```

redgate

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

07/07

Predefined Roles

redgate

Predefined Roles

- Cluster-level roles that can be granted
- `pg_read_all_data` (for example)
  - This means that if a role can `CONNECT` to a database, they can `SELECT` from all tables

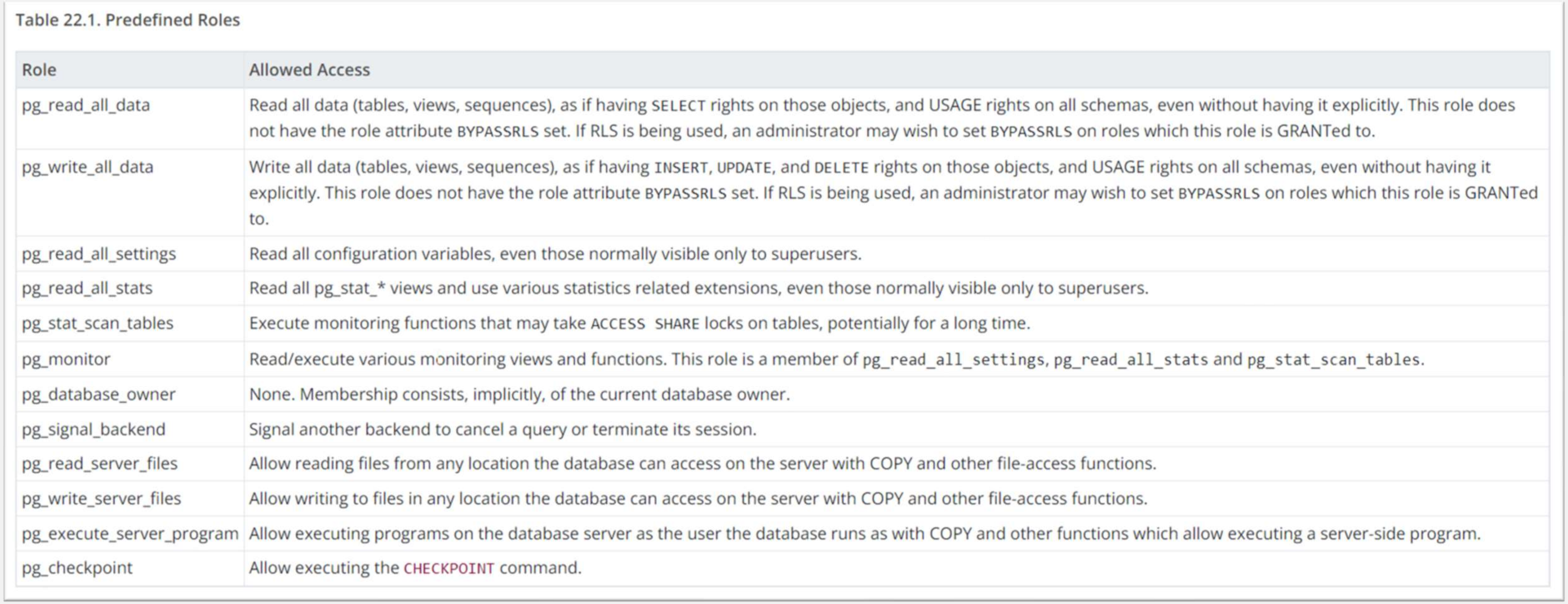


Table 22.1. Predefined Roles

Role	Allowed Access
pg_read_all_data	Read all data (tables, views, sequences), as if having SELECT rights on those objects, and USAGE rights on all schemas, even without having it explicitly. This role does not have the role attribute BYPASSRLS set. If RLS is being used, an administrator may wish to set BYPASSRLS on roles which this role is GRANTed to.
pg_write_all_data	Write all data (tables, views, sequences), as if having INSERT, UPDATE, and DELETE rights on those objects, and USAGE rights on all schemas, even without having it explicitly. This role does not have the role attribute BYPASSRLS set. If RLS is being used, an administrator may wish to set BYPASSRLS on roles which this role is GRANTed to.
pg_read_all_settings	Read all configuration variables, even those normally visible only to superusers.
pg_read_all_stats	Read all pg_stat_* views and use various statistics related extensions, even those normally visible only to superusers.
pg_stat_scan_tables	Execute monitoring functions that may take ACCESS SHARE locks on tables, potentially for a long time.
pg_monitor	Read/execute various monitoring views and functions. This role is a member of pg_read_all_settings, pg_read_all_stats and pg_stat_scan_tables.
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 COPY and other file-access functions.
pg_write_server_files	Allow writing to files in any location the database can access on the server with COPY and other file-access functions.
pg_execute_server_program	Allow executing programs on the database server as the user the database runs as with COPY and other functions which allow executing a server-side program.
pg_checkpoint	Allow executing the CHECKPOINT command.

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

What Questions do you have?

redgate

🎉 THANK YOU! 🎉

github.com/ryanbooz/presentations

redgate