Greenplum Roles, Privileges and Resources

Pivotal® Greenplum Database

Pivotal™

Agenda

- **Defining roles and privileges**
- Assigning Privileges to roles
- Securing Databases and data

Roles and Privileges Overview

Roles:

- Can be a user, group, or both
- Are not related to OS users and groups
- Use attributes to determine permission levels
- Are given access privileges to database objects
- Can be members of other roles
- Are defined at the system-level

Every system has a default superuser role

Role Privileges to Create Users

A user account:

- Has login privileges
- Is automatically assigned the following default attributes:
  - `NOSUPERUSER`
  - `NOCREATEDB`
  - `NOCREATEROLE`
  - `INHERIT`
  - `NOLOGIN` (must explicitly give `LOGIN` to user-level roles)

Roles – Superusers

A superuser:

- Bypasses all permission checks
- Should not be used for daily administration

Create the following administrative roles to work with:

- `SUPERUSER`
- `CREATEDB`
- `CREATEROLE`
- `PASSWORD`

Note: It is good practice to create a role that has the `CREATEDB` and `CREATEROLE` privileges, but is not a superuser. Use this role for all routine management of databases and roles. This approach avoids the dangers of operating as a superuser for tasks that do not require it.

Common Role Attributes

Role Attribute	Description
SUPERUSER | NOSUPERUSER	Determines if the role is a superuser. You must yourself be a superuser to create a new superuser. NOSUPERUSER is the default.
CREATEDB | NOCREATEDB	Determines if the role is allowed to create databases. NOCREATEDB is the default.
CREATEROLE | NOCREATEROLE	Determines if the role is allowed to create and manage other roles. NOCREATEROLE is the default.
INHERIT | NOINHERIT	Determines whether a role inherits the privileges of roles it is a member of. INHERIT is the default.
LOGIN | NOLOGIN	Determines whether a role is allowed to log in. NOLOGIN is the default.
CONNECTION LIMIT connlimit	If role can log in, this specifies how many concurrent connections the role can make. -1 (the default) means no limit.

Common Role Attributes (Cont)

Role Attribute	Description
PASSWORD 'password'	Sets the role's password. A null password can optionally be written explicitly as PASSWORD NULL.
ENCRYPTED | UNENCRYPTED	Controls whether the password is stored encrypted in the system catalogs. The default behavior is determined by the configuration parameter password_encryption (currently set to MD5).
VALID UNTIL 'timestamp'	**Sets a date and time after which the role's password is no longer valid. If omitted the password will be valid for all time.**
RESOURCE QUEUE queue_name	Assigns the role to the named resource queue for workload management.

SQL Commands for Roles

Use the following SQL commands and Greenplum command line application to manage roles:

Action	SQL Syntax	Greenplum Command Line Application
Create a role	CREATE ROLE	createuser
Drop a role	DROP ROLE	dropuser
Alter a role	ALTER ROLE	

Pivotal

Agenda

- Defining roles and privileges
- **Assigning Privileges to roles**
- Securing Databases and data

Roles and Privileges – Example

Example: Create roles with the `LOGIN` privilege

```
CREATE ROLE john WITH LOGIN;
CREATE USER john;
```

Example: Change a role and assign it `CREATEDB` privileges

```
ALTER ROLE john WITH CREATEDB;
```

Example: Grant read access to the `gphdfs` protocol

```
GRANT SELECT ON PROTOCOL gphdfs TO john
```

Role Membership or Groups

A role:

- Can be a member of other roles
- Inherits object privileges of the parent role
- Allows you to set object privileges in one place
- Will not inherit:
  - `LOGIN`
  - `SUPERUSER`
  - `CREATEDB`
  - `CREATEROLE`
- Can use `SET ROLE` to obtain privileges

Role Membership or Groups – Examples

To manage access to roles:

- Use `GRANT` command to grant membership
- Use `REVOKE` command to remove a member from a role

Example: Grant and revoke privileges

```
CREATE ROLE admin CREATEROLE CREATEDB;
GRANT admin TO john, sally;
REVOKE admin FROM bob;
SET ROLE admin;
```

Pivotal

Object Privileges

Objects:

- Are owned by object creator
- Can be made accessible to other roles
- Can be made accessible to all through the `PUBLIC` role
- Are managed with `GRANT` and `REVOKE` commands
- Assigned to deprecated roles are managed with `DROP OWNED` and `REASSIGN OWNED`

Database Object Privileges

Tables, Views, and Sequences	External Tables	Databases	Functions	Procedural Languages	Schemas
SELECT	SELECT	CREATE	EXECUTE	USAGE	CREATE
INSERT	RULE	CONNECT			USAGE
UPDATE	ALL	TEMPORARY			ALL
DELETE		ALL			
RULE					
TRUNCATE					
ALL					

Pivotal

Object Privileges – Examples

Example: Grant permissions to `admin`

```
GRANT ALL ON DATABASE
mydatabase TO admin
WITH GRANT OPTION;
```

Example: Assign all of one user's objects to another user

```
REASSIGN OWNED BY sally TO bob;
```

Example: Grant `SELECT` to `PUBLIC`

```
GRANT SELECT ON TABLE
mytable TO PUBLIC;
```

Example: Drop all objects owned by `visitor`

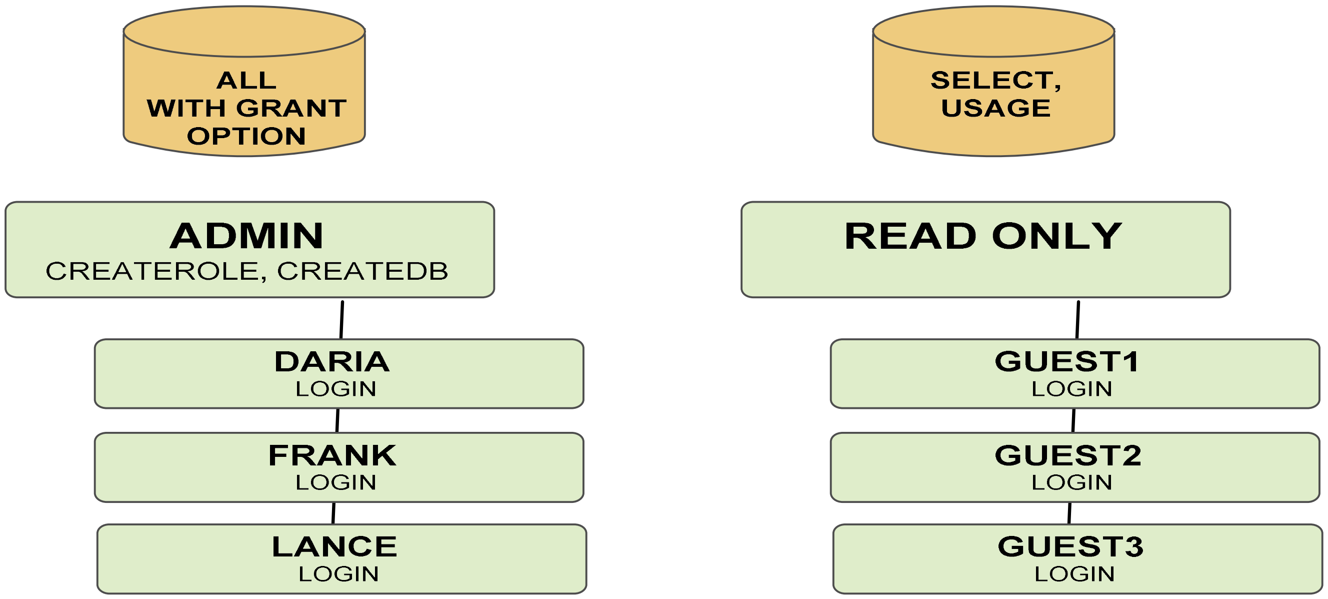
```
DROP OWNED BY visitor;
```

Example: Remove `INSERT` and `UPDATE`

```
REVOKE INSERT UPDATE ON TABLE
mytable FROM sally;
```

Pivotal

Role Structure Example



Agenda

- Defining roles and privileges
- Assigning Privileges to roles
- **Securing Databases and data**

System-level Security

Problems encountered with system security:

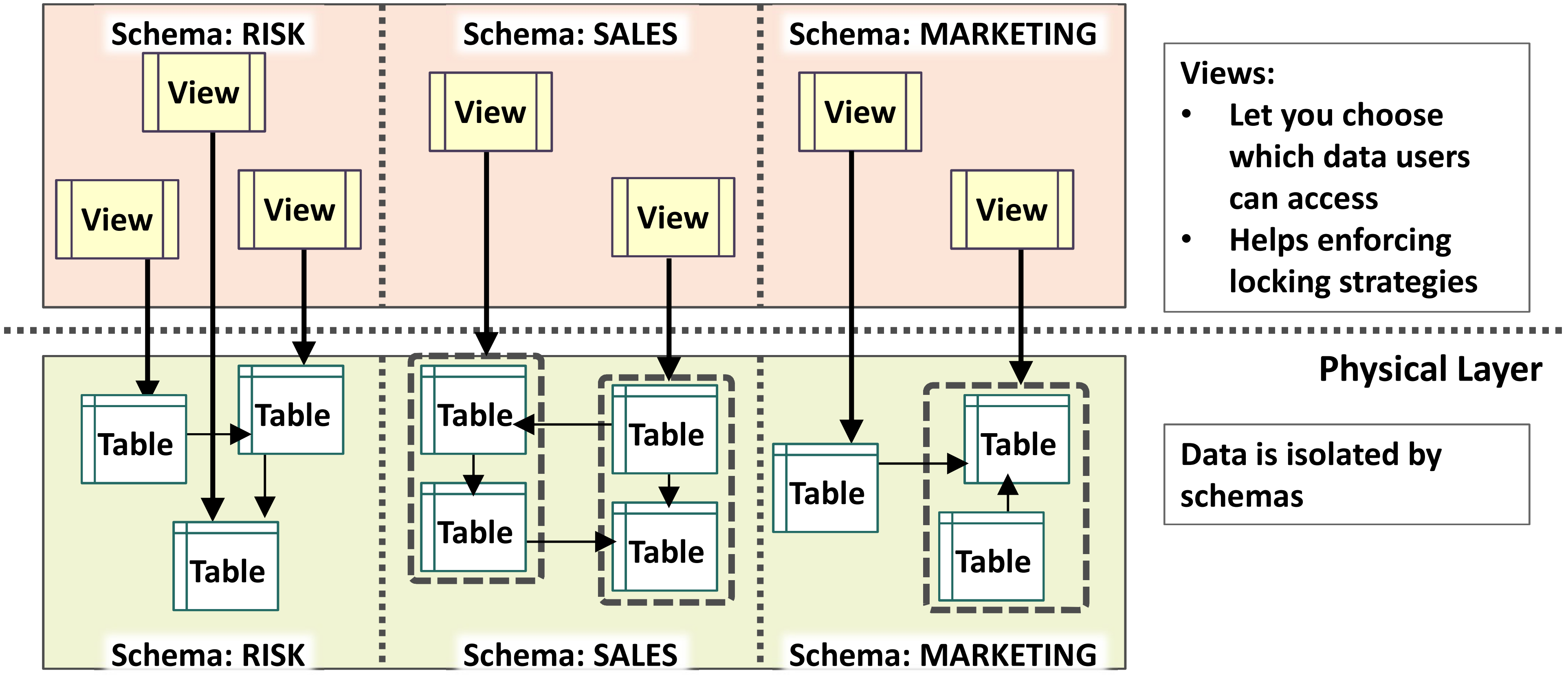
- Pessimistic model is used, so users do not have access to anything unless specifically assigned.
- Sharing accounts makes auditing difficult
- Users are tempted to run additional workloads on master node, such as ETL
- Once logged into the master, all other hosts are available since SSH keys are exchanged
- Network infrastructure is not under our control
- System administration is not under our control

Database Security – Roles

Roles can be used to enhance security with the following:

- Each person should have their own role
- Roles should be further divided into groups, which are usually roles that do not have the `LOGIN` attribute
- Privileges should be granted at the group level whenever possible
- Privileges should be as restrictive as possible
- Column level access can be accomplished with views
- Roles are not related to OS users or groups

Database Architecture – Separation and Isolation



Views:
- Let you choose which data users can access
- Helps enforcing locking strategies

Data is isolated by schemas

Security Example

The following is an example of user and role creation and assignment and how to use the roles to limit or grant privileges to user roles:

Example: Inherit privileges through nested roles

```
CREATE ROLE batch;
GRANT select, insert, update, delete
   ON dimensions.customer TO batch;
CREATE ROLE batchuser LOGIN;
GRANT batch TO batchuser;
```

Additional Security Considerations

Two partners that provide additional security

Zettaset

- encrypted network
- encrypted drive capability
- Protects against people eavesdropping on the network
- Or if someone steals the machine the disk can’t be read

Protegrity

- Role based encryption
- encrypted columns and roles and table;
- only specific users can read it
- even super users and gpadmins cannot see what is encrypted

Wrapping Up

- Roles and privileges
- Role assignment to object privileges
- Security issues that can affect your database and corresponding data
- Creating roles with specific privileges to control access
- Isolating users at the physical and functional layers