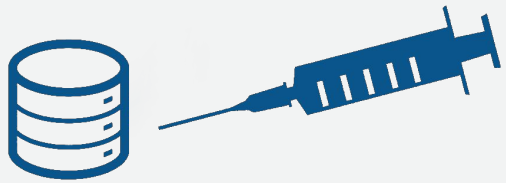
# SECURITY

## DATABASE MANAGEMENT SYSTEM

Sujan Tamrakar

At some point in 2012, according to X-Force research, SQL injection attacks were responsible for more than half of all data breaches where the attack type had been disclosed. While that number has decreased in 2015, it is still one of the main attack vectors being used.

In 2008, cybercriminals made off with information from as many as 100 million debit and credit cards in the Heartland Payment Systems data breach. At the time, it was the largest data breach ever, and SQL injection was the culprit.



SQL Injection

## SQL Injection.

User-Id : jashwanth

Password : newpassword

```
select * from Users where user_id= ' jashwanth '
                      and password = ' newpassword '
```

User-Id : ' OR 1= 1; /*

Password : */--

```
select * from Users where user_id= '' OR 1 = 1; /* '
                      and password = ' */-- '
```

# SECURITY

- Db contains crucial information of any system so, it should be secured against variety of threats.
- Not every user of db system should be able to access all the data.
- Data, db application, stored functions/procedures, etc. should be protected against compromises of their Confidentiality, Integrity & Availability (CIA).


# SECURITY RISKS

- Misuse by unauthorized db users or unauthorized users.
- Infections causing unauthorized access, data leakage, disclosure of personal data, damage to data or programs, interruption or denial of access to db, attack on other systems, unanticipated failure of db service, overloads, performance constraints, etc.
- Physical damage to db servers caused by natural disasters (fire, earthquake, volcano, flood), over-heating, lightning, static discharge, equipment failure, etc.
- Design flaws & programming bugs in db creating various security vulnerabilities, loopholes, data loss, corruption, performance degradation, etc.
- Data corruption and/or loss caused by entry of invalid data or commands, mistakes in db administration process, lacking in data validation.

# SECURITY RISKS

Types of information security control for any database system:

- Access control
- Auditing
- Authentication
- Encryption
- Integrity controls
- Backups
- Application security

# ACCESS CONTROL

- Selective restriction of access to a place or other resources.
- Accessing = consuming, entering, using

**Physical Security:**

- Refers to practice of restricting entrance to a property (building/room/office) to unauthorized users. Only allows to enter authenticated users.
- Can be achieved by human, mechanical means like keys, locks, or any technological means of barring.



# ACCESS CONTROL


**Logical Security:**

- General access control includes authentication, authorization, access approval & audit.
- Authentication = verifies who you are [passwords, biometric scans, physical/electronic keys], only legitimate users can login
- Authorization = what you are authorized (approved) to do.
- Access approval = grant access to user & associate with some suitable resources
- Audit = monitoring, assessing, recording user's actions
- Accountability = what a subject did


# ACCESS CONTROL

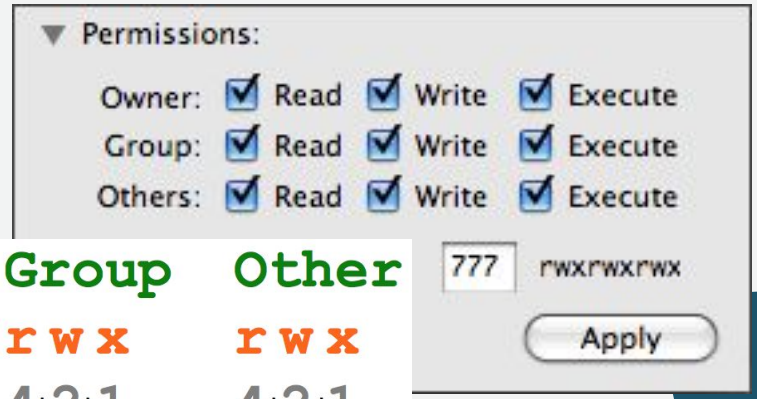
Access control models tend to fall into one of 2 classes;

1. Based on capabilities:
    - Holding an unforgettable reference to an object provides access to the object.
    - Access is conveyed to another party by transmitting such a capability over a secure channel
    - [House key grants to access house]
2. Based on Access Control List (ACL):
    - A subject's access to an object depends on whether its identity is on a list associated with the object.
    - [Checking of ID to enter an area]

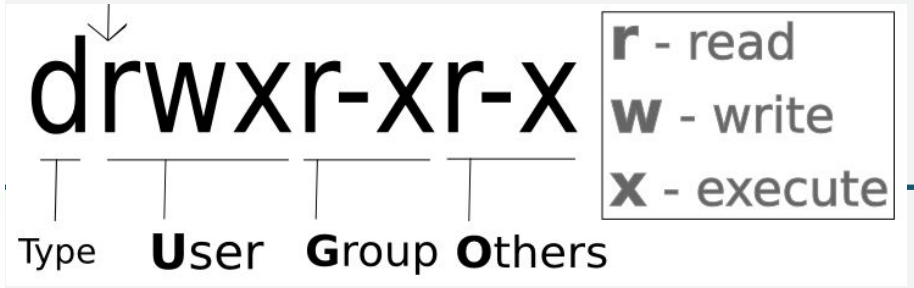
# DATABASE SECURITY & DBA

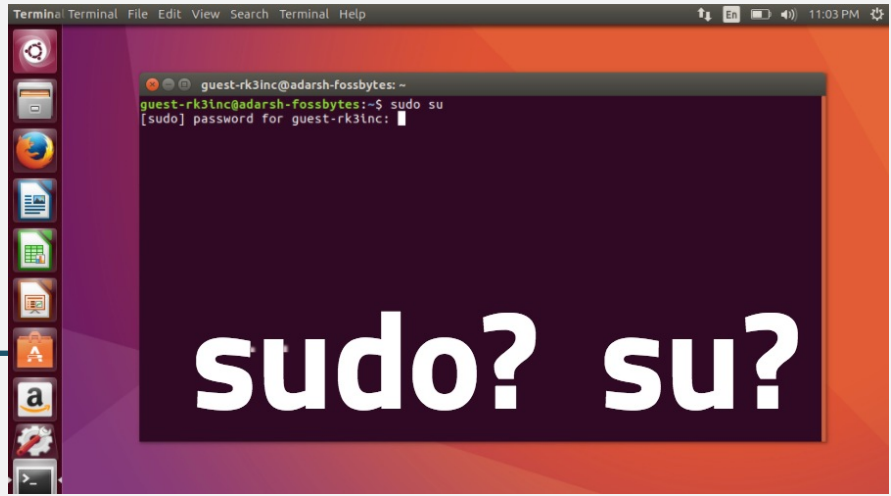
- ✔ DBA (Database Administrator) is the central authority for managing a db system & is responsible for db management.
- ✔ DBA's responsibilities include granting privileges to users/groups, revoking privileges, classifying users & data according to organization's policy, creating accounts, managing security layers, etc.
- ✔ DBA's account is called Superuser account which provides powerful capabilities that are not made available to regular accounts.

# AUTHORIZATION




- ✔ Authorization to information as well as schema like:
  - ✔ Create new relation
  - ✔ Add attributes or delete attributes from a relation
  - ✔ Drop a relation
- ✔ Such schema level operations are only allowed to modify by Superuser unless assigned to other accounts. Ultimate form of authority holds with Superuser i.e. DBA.
- ✔ Only those user who has the execute privileges can call and run the functions, procedures.

# AUTHORIZATION

- Granting privileges:
  - ✔ A user who has been granted some form of authorization may be allowed to pass on this authorization to other users.
  - ✔ This passed authorization should be able to cancel at any time.

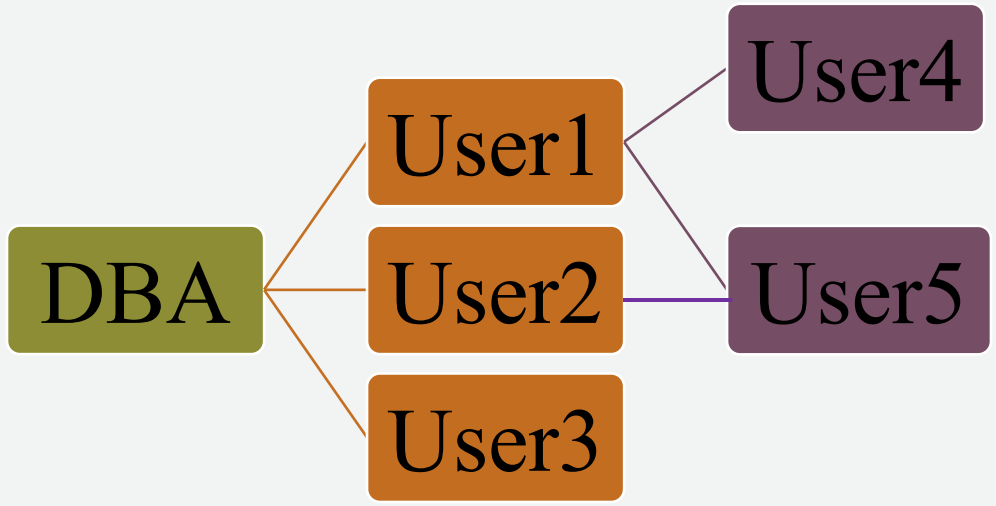



Fig. Authorization grant graph

# AUTHORIZATION

- Granting privileges in SQL:

  Syntax:

  grant <privilege list> on <relation name / view name> to <user / role>

  Example:

  grant select on account to U1, U2, U3;

  grant update (amount) on loan to U1, U2;

  grant select on branch to U1 with grant option;

# AUTHORIZATION

**Roles**

- Help to assign a set of privileges to a user according to the role that the user plays in the organization.
- Authorizations are granted to roles like granted to any user.
- Example of roles:
  - teller, branchManager, auditor, sysAdmin
  - Student, teacher, librarian, accountant, vicePrincipal, sysadmin

# AUTHORIZATION

## Roles

- SQL examples (granting): - permit an authorization
  - Create role teller;
  - Grant select on account to teller; (teller assigned with select command on account table)
  - Grant teller to john; (‘teller’ role granted to john)
  - Create role manager;
  - Grant manager to mary;
- SQL examples (revoking): - cancel the authorization
  - Syntax: revoke <privilege list> on <relation / view name> from <user/role list> [restrict|cascade]
  - Example: revoke select on branch from U1, U2, U3;
    revoke update (amount) on loan from U1, U2, U3;

# Thank you