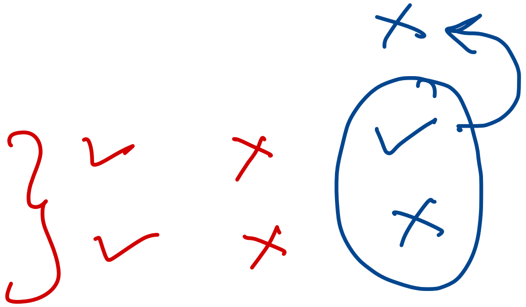
MySQL - RDBMS

Trainer: Mr. Nilesh Ghule

Transaction

- Transaction is set of DML queries executed as a single unit.
- Transaction examples
  - accounts table [id, type, balance]
  - UPDATE accounts SET balance=balance-1000 WHERE id = 1;
  - UPDATE accounts SET balance=balance+1000 WHERE id = 2;



- RDBMS transaction have ACID properties.
  - Atomicity
    - All queries are executed as a single unit. If any query is failed, other queries are discarded.
  - Consistency
    - When transaction is completed, all clients see the same data.
  - Isolation
    - Multiple transactions (by same or multiple clients) are processed concurrently.
  - Durable
    - When transaction is completed, all data is saved on disk.

Transaction

- Transaction management
  - START TRANSACTION;
  - …
  - COMMIT WORK;

  - START TRANSACTION;
  - …
  - ROLLBACK WORK;
- In MySQL autocommit variable is by default 1. So each DML command is auto-committed into database.
  - SELECT @@autocommit;
- Changing autocommit to 0, will create new transaction immediately after current transaction is completed. This setting can be made permanent in config file.
  - SET autocommit=0;

Views

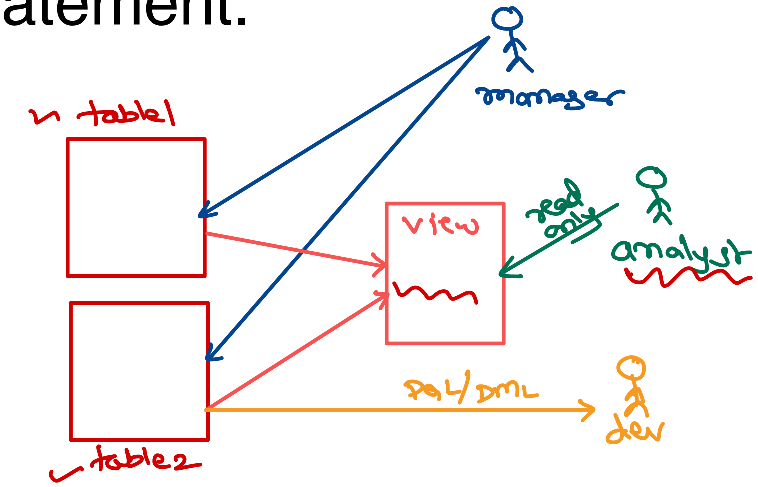
- RDBMS view represents view (projection) of the data.
- View is based on SELECT statement.
- Typically it is restricted view of the data (limited rows or columns) from one or more tables (joins and/or sub-queries) or summary of the data (grouping).
- Data of view is not stored on server hard-disk; but its SELECT statement is stored in compiled form. It speed up execution of view.

Views

- Views are of two types: Simple view and Complex view
- Usually if view contains computed columns, group by, joins or sub-queries, then the views are said to be complex. DML operations are not supported on these views.
- DML operations on view affects underlying table.
- View can be created with CHECK OPTION to ensure that DML operations can be performed only the data visible in that view.

View

- Views can be differentiated with: SHOW FULL TABLES.
- Views can be dropped with DROP VIEW statement.
- View can be based on another view.



- Applications of views
  - Security: Providing limited access to the data.
  - Hide source code of the table.
  - Simplifies complex queries.

example joins/subqueries are added into view - select.

Data Control Language

- Security is built-in feature of any RDBMS. It is implemented in terms of permissions (a.k.a. privileges).
- There are two types of privileges.
- System privileges

  database, user

  - Privileges for certain commands i.e. CREATE, ALTER, DROP, ...
  - Typically these privileges are given to the database administrator or higher authority user.

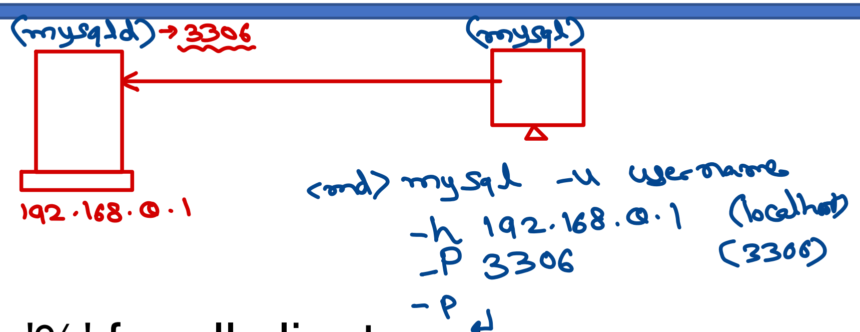
  MySQL → root user
- Object privileges
  - RDBMS objects are table, view, stored procedure, function, triggers, …
  - Can perform operations on the objects i.e. INSERT, UPDATE, DELETE, SELECT, CALL, ...
  - Typically these privileges are given to the database users.

User Management

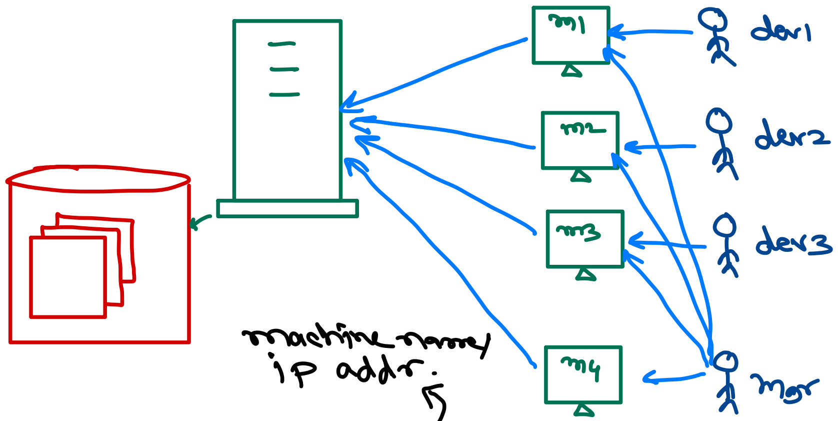
- User management is responsibility of admin (root).
- New user can be created using CREATE USER.
  - CREATE USER user@host IDENTIFIED BY 'password';
  - host can be hostname of server, localhost (current system) or '%' for all client systems.
- Permissions for the user can be listed using SHOW GRANTS command.
  - SHOW GRANTS FOR user@host;
- Users can be deleted using DROP USER.
  - DROP USER 'user'@'host';
- Change user password.
  - ALTER USER user@host IDENTIFIED BY 'new_password';
  - FLUSH PRIVILEGES; → To reload security settings if they are modified using DML ops on system tables.

In RDBMS, all configs/settings/metadata is stored in system tables/databases.

mysql.user
db table



```
cmd> mysql -u username
  -h 192.168.0.1   (localhost)
  -P 3306          (3306)
  -p ↵
```



machine name/
ip addr.

```
create user 'dev1'@'m1' identified by '...';
create user 'dev2'@'m2' ____ '' ________
create user 'mgr'@'%' ____ '' ________
```

↑ any machine.

Data Control Language

everything in db.

- Permissions are given to user using GRANT command.
  - GRANT CREATE ON db.* TO user@host;
  - GRANT CREATE ON *.* TO user1@host, ~~user2@host~~;
  - GRANT SELECT ON db.table TO user@host;
  - GRANT SELECT, INSERT, UPDATE ON db.table TO user@host;
  - GRANT ALL ON db.* TO user@host;
- By default one user cannot give permissions to other user. This can be enabled using WITH GRANT OPTION.

  system priv (all db)

  - GRANT ALL ON *.* TO user@host WITH GRANT OPTION;
- Permissions assigned to any user can be withdrawn using REVOKE command.
  - REVOKE SELECT, INSERT ON db.table FROM user@host;
- Permissions can be activated by FLUSH PRIVILEGES.
  - System GRANT tables are reloaded by this command. Auto done after GRANT, REVOKE.
  - Command is necessary i~~s~~f GRANT tables are modified using DML operations.

Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>