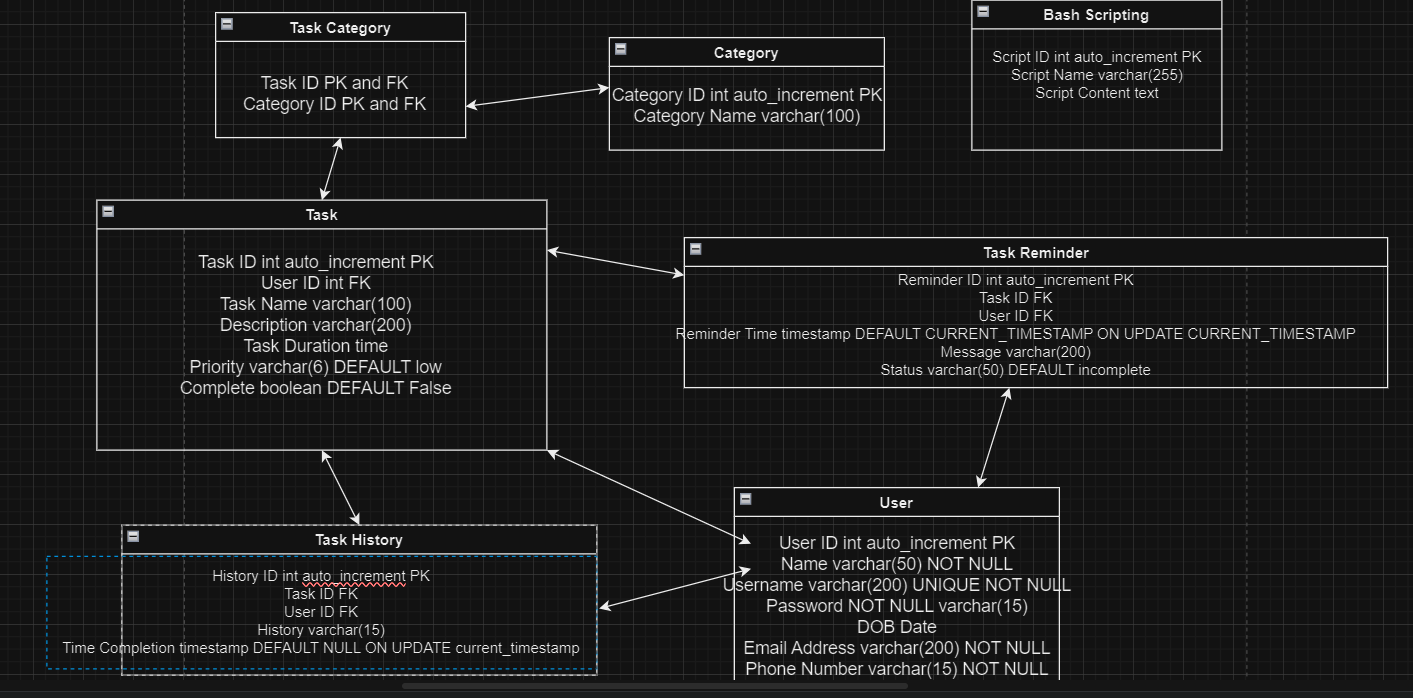
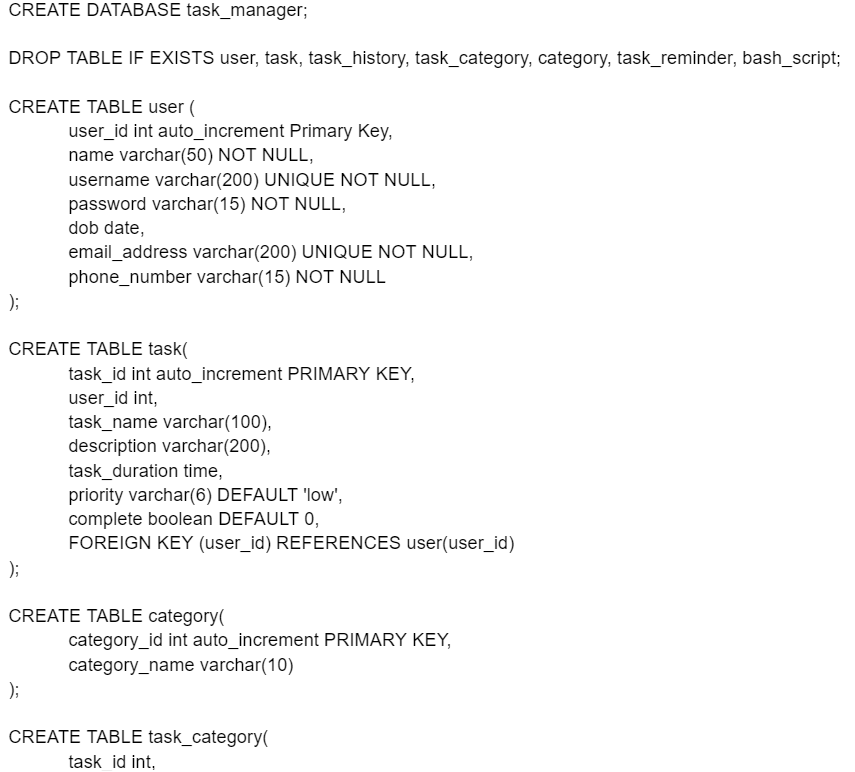
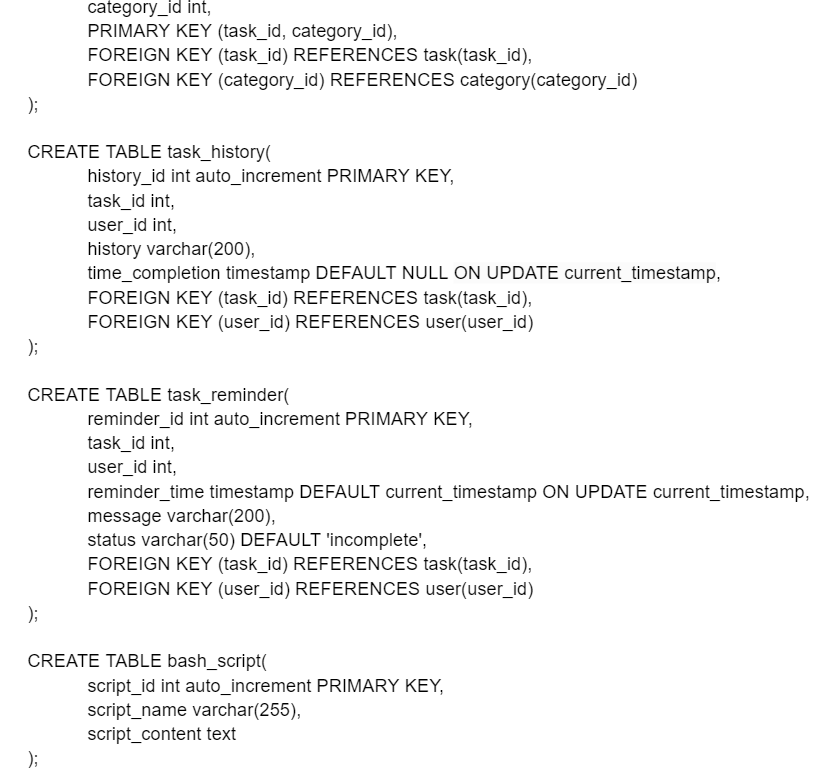
**Schema and Tables**

****

**DDL COMMANDS**

****

****

**Stored Procedures**

**Creating a new user**

DELIMETER//

CREATE PROCEDURE create\_user(

-> IN name VARCHAR(50),

-> IN username VARCHAR(200),

-> IN password VARCHAR(64), -- Change the length to match the hashed password length

-> IN dob DATE,

-> IN email\_address VARCHAR(200),

-> IN phone\_number VARCHAR(15)

-> )

-> BEGIN

-> INSERT INTO user (name, username, password, dob, email\_address, phone\_number)

-> VALUES (name, username, password, dob, email\_address, phone\_number);

-> END //

**Updating an existing profile**

DELIMITER //

CREATE PROCEDURE update\_profile(

IN name varchar(50),

IN username varchar(200),

IN password varchar(15),

IN dob date,

IN email\_address varchar(200),

IN phone\_number varchar(15)

)

BEGIN

UPDATE user

SET

name = name,

username = username,

password = password,

dob = dob,

email\_address = email\_address,

phone\_number = phone\_number

WHERE user\_id = user\_id;

END //

**Creating a new task**

DELIMITER //

CREATE PROCEDURE new\_task(

IN task\_name varchar(100),

IN description varchar(200),

IN task\_duration time,

IN priority varchar(6),

IN complete boolean

)

BEGIN

INSERT INTO task (task\_name, description, task\_duration, priority, complete)

VALUES (task\_name, description, task\_duration, priority, complete);

END //

**Updating a task**

DELIMITER //

CREATE PROCEDURE update\_task(

IN task\_id int,

IN task\_name varchar(100),

IN description varchar(200),

IN task\_duration time,

IN priority varchar(6),

IN complete boolean

)

BEGIN

UPDATE task

SET

task\_name = task\_name,

description = description,

task\_duration = task\_duration,

priority = priority,

complete = complete

WHERE task\_id = task\_id;

END //

**List all task that aren't complete**

DELIMITER //

CREATE PROCEDURE list\_incomplete\_task(

IN user\_id int

)

BEGIN

SELECT \*

FROM task

WHERE user\_id = user\_id AND complete = 0;

END //

**List all task that are complete**

DELIMITER //

CREATE PROCEDURE list\_complete\_task(

IN user\_id int

)

BEGIN

SELECT \*

FROM task

WHERE user\_id = user\_id AND complete = 1;

END //

**Trigger for task\_history**

DELIMITER //

CREATE TRIGGER task\_history\_trigger

AFTER INSERT ON task

FOR EACH ROW

BEGIN

INSERT INTO task\_history (task\_id, user\_id, history, time\_completion)

VALUES (NEW.task\_id, NEW.user\_id, CONCAT('Task created: ', NEW.task\_name), NULL);

END //

**Creating a reminder**

DELIMITER //

CREATE PROCEDURE create\_reminder(

IN task\_id INT,

IN user\_id INT,

IN reminder\_time TIMESTAMP,

IN message VARCHAR(200),

IN status VARCHAR(50)

)

BEGIN

INSERT INTO task\_reminder (task\_id, user\_id, reminder\_time, message, status)

VALUES (task\_id, user\_id, reminder\_time, message, status);

END //

**Updating reminder**

DELIMITER //

CREATE PROCEDURE update\_reminder(

IN reminder\_id int,

IN reminder\_time timestamp,

IN message varchar(200),

IN status varchar(50)

)

BEGIN

UPDATE task\_reminder

SET

reminder\_time = reminder\_time,

message = message,

status = status

WHERE reminder\_id = reminder\_id;

END //

**Deleting a Reminder**

DELIMITER //

CREATE PROCEDURE delete\_reminder(

IN reminder\_id int

)

BEGIN

DELETE FROM task\_reminder WHERE reminder\_id = reminder\_id;

END //

**Create a category name**

DELIMITER //

CREATE PROCEDURE create\_category(

IN category\_name varchar(10)

)

BEGIN

INSERT INTO category (category\_name)

VALUES (category\_name);

END //

**Deleting a task**

DELIMITER //

CREATE PROCEDURE delete\_task(

IN task\_id int

)

BEGIN

DELETE FROM task WHERE task\_id = task\_id;

END //

**Pulling task details**

DELIMITER //

CREATE PROCEDURE get\_user\_task\_details(

IN user\_id int

)

BEGIN

SELECT

u.name,

t.task\_name,

t.task\_duration,

th.time\_completion,

tr.reminder\_time,

c.category\_name

FROM

user u

JOIN

task t ON u.user\_id = t.user\_id

LEFT JOIN

task\_history th ON t.task\_id = th.task\_id

LEFT JOIN

task\_reminder tr ON t.task\_id = tr.task\_id

LEFT JOIN

task\_category tc ON t.task\_id = tc.task\_id

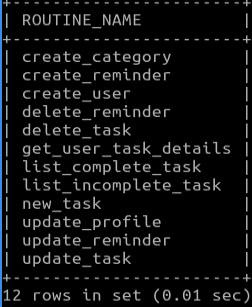
LEFT JOIN

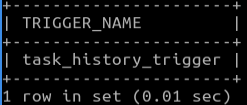
category c ON tc.category\_id = c.category\_id

WHERE

u.user\_id = p\_user\_id;

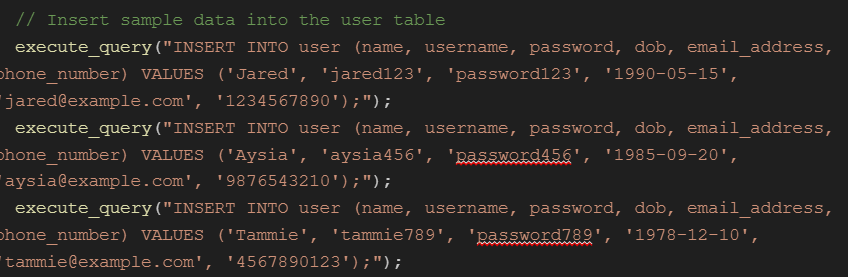
END //

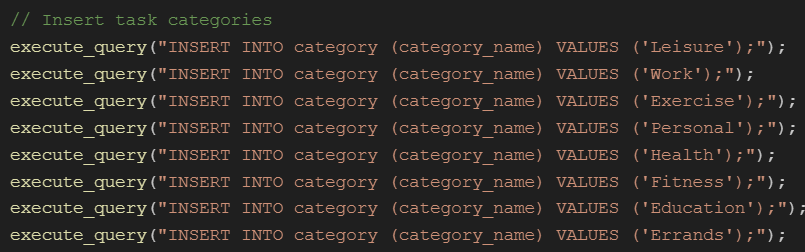
****

****

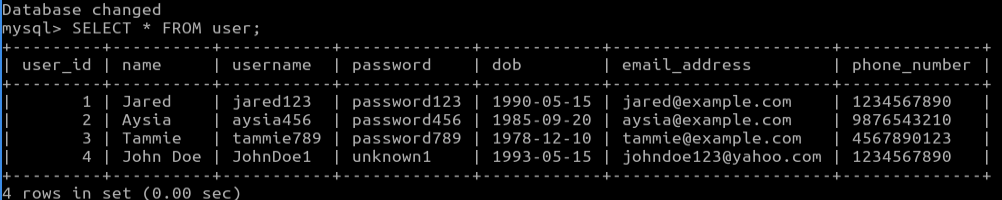
**DML Commands**

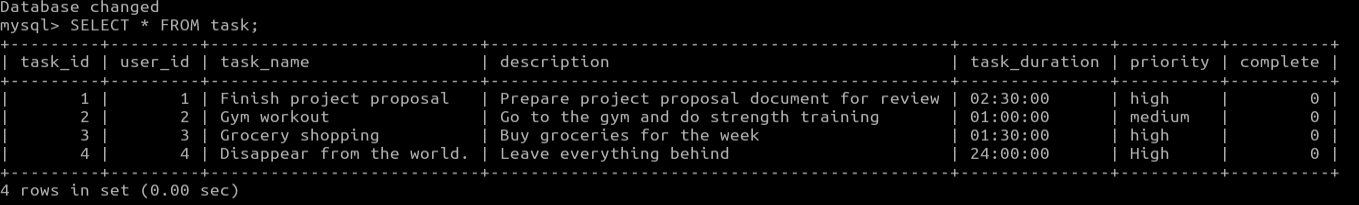
**Queries**

****

****

**Results**

****

****

****

****