```sql
-- Demonstrates granting and revoking privileges in MySQL
-- Uses `rideshare` database

-- Creates a new user with username 'carter' and password 'password'
CREATE USER 'carter' IDENTIFIED BY 'password';

-- As Carter, fails to show databases
SHOW DATABASES;

-- As root user, grants SELECT privileges on only the `analysis` view in the `rideshare` database
GRANT SELECT ON `rideshare`.`analysis` TO 'carter';

-- As Carter, succeeds in showing `rideshare` database
SHOW DATABASES;
USE `rideshare`;

-- As Carter, succeeds in selecting from `analysis`
SELECT * FROM `analysis`;

-- As Carter, fails to select from `rides` table
SELECT * FROM `rides`;

-- As Carter, fails to create a new view
CREATE VIEW `destinations` AS
SELECT `destination` FROM `analysis`;

-- As root user, grants privileges to 'carter' to create a view on any table in the `ridershare` database
GRANT CREATE VIEW ON `rideshare`.* TO 'carter';

-- As Carter, succeeds in creating a view
CREATE VIEW `destinations` AS
SELECT `destination` FROM `analysis`;
```

```sql
-- Demonstrates schema for MySQL `rideshare` database

CREATE DATABASE IF NOT EXISTS `rideshare`;
USE `rideshare`;

CREATE TABLE `rides` (
    `id` INT AUTO_INCREMENT,
    `origin` VARCHAR(64) NOT NULL,
    `destination` VARCHAR(64) NOT NULL,
    `rider` VARCHAR(16) NOT NULL,
    PRIMARY KEY(`id`)
);

INSERT INTO `rides` (`origin`, `destination`, `rider`)
VALUES
('Good Egg Galaxy', 'Honeyhive Galaxy', 'Peach'),
('Castle Courtyard', 'Cascade Kingdom', 'Mario'),
('Metro Kingdom', 'Mushroom Kingdom', 'Luigi'),
('Seaside Kingdom', 'Deep Woods', 'Bowser');

CREATE VIEW `analysis` AS
SELECT `id`, `origin`, `destination`
FROM `rides`;
```

```sql
-- Demonstrates SQL injection attacks
-- Uses `bank` database

-- Executes statement with non-malicious input
SELECT * FROM `accounts`
WHERE `id` = 1;

-- Executes statement with malicious input
SELECT * FROM `accounts`
WHERE `id` = 1 UNION SELECT * FROM `accounts`;
```

```sql
-- Demonstrates prepared statements
-- Uses `bank` database

-- Creates a prepared statement
PREPARE `balance_check`
FROM 'SELECT * FROM `accounts`
WHERE `id` = ?';

-- Executes the prepared statement with non-malicious input
SET @id = 1;
EXECUTE `balance_check` USING @id;

-- Executes the prepared statement with malicious input
SET @id = '1 UNION SELECT * FROM `accounts`';
EXECUTE `balance_check` USING @id;
```

```sql
-- Demonstrates schema for MySQL `bank` database

CREATE DATABASE IF NOT EXISTS `bank`;
USE `bank`;

CREATE TABLE `accounts` (
    `id` INT AUTO_INCREMENT,
    `name` VARCHAR(16),
    `balance` INT,
    PRIMARY KEY(`id`)
);

INSERT INTO `accounts` (`name`, `balance`)
VALUES
('Alice', 10),
('Bob', 20),
('Charlie', 30);
```

```
-- Demonstrates navigating a MySQL database

-- Starts MySQL server using Docker
docker container run --name mysql -p 3306:3306 -v /workspaces/$RepositoryName:/mnt -e MYSQL_ROOT_PASSWORD=crimson -d mysql

-- Logs in, if using 'crimson' as password
mysql -h 127.0.0.1 -P 3306 -u root -p

-- Lists all databases on server
SHOW DATABASES;
```

```sql
-- Demonstrates stored procedures
-- Uses `mfa` database

-- Sets up database
USE `mfa`;

-- Adds deleted column to columns
ALTER TABLE `collections`
ADD COLUMN `deleted` TINYINT DEFAULT 0;

-- Creates a stored procedure to view all (non-deleted) items in collections
delimiter //
CREATE PROCEDURE `current_collections`()
BEGIN
SELECT `title`, `accession_number`, `acquired` FROM `collections` WHERE `deleted` = 0;
END//
delimiter ;

-- Calls the stored procedure
CALL `current_collections`();

-- Sets an item to be deleted
UPDATE `collections` SET `deleted` = 1
WHERE `title` = 'Farmers working at dawn';

-- Calls stored procedure again
CALL `current_collections`();

-- Creates a table to log artwork transactions
CREATE TABLE `transactions` (
    `id` INT AUTO_INCREMENT,
    `title` VARCHAR(64) NOT NULL,
    `action` ENUM('bought', 'sold') NOT NULL,
    PRIMARY KEY(`id`)
);

-- Creates a stored procedure with a parameter to mark artwork sold
delimiter //
CREATE PROCEDURE `sell`(IN `sold_id` INT)
BEGIN
UPDATE `collections` SET `deleted` = 1 WHERE `id` = `sold_id`;
INSERT INTO `transactions` (`title`, `action`)
```

```
VALUES ((SELECT `title` FROM `collections` WHERE `id` = `sold_id`), 'sold');
END//
delimiter ;

-- Sells a piece of artwork
CALL `sell`((
    SELECT `id` FROM `collections`
    WHERE `title` = 'Farmers working at dawn'
));

-- Shows results
SELECT * FROM `transactions`;
SELECT * FROM `collections`;

-- Delete procecure to later improve it
DROP PROCEDURE `sell`;

-- Creates a stored procedure to handle case where item is already deleted
delimiter //
CREATE PROCEDURE `sell`(IN `sold_id` INT)
BEGIN
IF `sold_id` IN (
    SELECT `id` FROM `collections` WHERE `deleted` = 0
) THEN
    UPDATE `collections` SET `deleted` = 1 WHERE `id` = `sold_id`;
    INSERT INTO `transactions` (`title`, `action`)
    VALUES ((SELECT `title` FROM `collections` WHERE `id` = `sold_id`), 'sold');
END IF;
END//
delimiter ;
```

```sql
-- Demonstrates schema for MySQL `mfa` database

CREATE DATABASE IF NOT EXISTS `mfa`;
USE `mfa`;

CREATE TABLE `collections` (
    `id` INT AUTO_INCREMENT,
    `title` VARCHAR(64) NOT NULL,
    `accession_number` VARCHAR(9) NOT NULL UNIQUE,
    `acquired` DATE,
    PRIMARY KEY(`id`)
);

INSERT INTO `collections` (`title`, `accession_number`, `acquired`)
VALUES
('Farmers working at dawn', '11.6152', '1911-08-03'),
('Imaginative landscape', '56.496', NULL),
('Profusion of flowers', '56.257', '1956-04-12'),
('Spring outing', '14.76', '1914-01-08');

CREATE TABLE `artists` (
    `id` INT AUTO_INCREMENT,
    `name` VARCHAR(64) NOT NULL,
    PRIMARY KEY(`id`)
);

INSERT INTO `artists` (`name`)
VALUES
('Li Yin'),
('Qian Weicheng'),
('Unidentified artist'),
('Zhou Chen');

CREATE TABLE `created` (
    `artist_id` INT,
    `collection_id` INT,
    PRIMARY KEY(`artist_id`, `collection_id`),
    FOREIGN KEY(`artist_id`) REFERENCES `artists`(`id`),
    FOREIGN KEY(`collection_id`) REFERENCES `collections`(`id`)
);

INSERT INTO `created` (`artist_id`, `collection_id`)
```

```
43  VALUES
44  ((SELECT `id` FROM `artists` WHERE `name` = 'Li Yin'), (SELECT `id` FROM `collections` WHERE `title` = 'Imaginative landscape')),
45  ((SELECT `id` FROM `artists` WHERE `name` = 'Qian Weicheng'), (SELECT `id` FROM `collections` WHERE `title` = 'Profusion of flowers')),
46  ((SELECT `id` FROM `artists` WHERE `name` = 'Unidentified artist'), (SELECT `id` FROM `collections` WHERE `title` = 'Farmers working at
dawn')),
47  ((SELECT `id` FROM `artists` WHERE `name` = 'Zhou Chen'), (SELECT `id` FROM `collections` WHERE `title` = 'Spring outing'));
```

```sql
-- Demonstrates expanded ALTER TABLE support in MySQL compared to SQLite
-- https://www.sqlite.org/omitted.html
-- Uses `mbta` database

-- Adds a new MBTA line
ALTER TABLE `stations`
MODIFY `line` ENUM('blue', 'green', 'orange', 'red', 'silver') NOT NULL;
```

```sql
-- Demonstrates types in MySQL and schema for `mbta` database

-- Creates and uses `mbta` database
CREATE DATABASE IF NOT EXISTS `mbta`;
USE `mbta`;

-- Creates tables with MySQL types
CREATE TABLE `cards` (
    `id` INT AUTO_INCREMENT,
    PRIMARY KEY(`id`)
);

DESCRIBE `cards`;

CREATE TABLE `stations` (
    `id` INT AUTO_INCREMENT,
    `name` VARCHAR(32) NOT NULL UNIQUE,
    `line` ENUM('blue', 'green', 'orange', 'red') NOT NULL,
    PRIMARY KEY(`id`)
);

DESCRIBE `stations`;

CREATE TABLE `swipes` (
    `id` INT AUTO_INCREMENT,
    `card_id` INT,
    `station_id` INT,
    `type` ENUM('enter', 'exit', 'deposit') NOT NULL,
    `datetime` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    `amount` DECIMAL(5,2) NOT NULL CHECK(`amount` != 0),
    PRIMARY KEY(`id`),
    FOREIGN KEY(`station_id`) REFERENCES `stations`(`id`),
    FOREIGN KEY(`card_id`) REFERENCES `cards`(`id`)
);

DESCRIBE `swipes`;

-- Shows all tables in `mbta` database
SHOW TABLES;
```

```
-- Demonstrates navigating a PostgreSQL database

-- Starts PostgreSQL server using Docker
docker run --name postgres -p 5432:5432 -v /workspaces/$RepositoryName:/mnt -e POSTGRES_PASSWORD=crimson -d postgres

-- Logs in, if using 'crimson' as password
psql postgresql://postgres@127.0.0.1:5432/postgres

-- Lists all databases
\l
\list

-- Creates a database
CREATE DATABASE "mbta";

-- Connects to a particular database
\c
\connect

-- Lists all tables
\dt

-- Describes a particular table
\d "cards"

-- Quits
\q
```

```sql
--- Demonstrates PostgreSQL types

CREATE TABLE "cards" (
    "id" SERIAL,
    PRIMARY KEY("id")
);

CREATE TABLE "stations" (
    "id" SERIAL,
    "name" VARCHAR(32) NOT NULL UNIQUE,
    "line" VARCHAR(32) NOT NULL,
    PRIMARY KEY("id")
);

CREATE TYPE "swipe_type" AS ENUM('enter', 'exit', 'deposit');

CREATE TABLE "swipes" (
    "id" SERIAL,
    "card_id" INT,
    "station_id" INT,
    "type" "swipe_type" NOT NULL,
    "datetime" TIMESTAMP NOT NULL DEFAULT now(),
    "amount" NUMERIC(5,2) NOT NULL CHECK("amount" != 0),
    PRIMARY KEY("id"),
    FOREIGN KEY("station_id") REFERENCES "stations"("id"),
    FOREIGN KEY("card_id") REFERENCES "cards"("id")
);
```