```
-- Creating database and using it
DROP DATABASE IF EXISTS winkit_demo;
CREATE DATABASE winkit_demo;
USE winkit_demo;

-- Creating Supplier table
CREATE TABLE Supplier(
   Supplier_id int PRIMARY KEY,
   name varchar(255),
   contact char(10) UNIQUE
);

-- Inserting data into Supplier table
INSERT INTO Supplier(Supplier_id, name, contact)
VALUES
(1,'P&G','9998874150'),
(2,'Hindustan Unilever','9998874151'),
(3,'Nestle','9998874152'),
(4,'Haldirams','9998874153'),
(5,'Dettol','9998874154'),
(6,'Dove','9998874155'),
(7,'Nivea','9998874156'),
(8,'Coca-Cola','9998874157'),
(9,'Goldie Suppliers','9998874158'),
(10,'Mankind','9998874159');

-- ALTER TABLE Supplier
-- ADD CONSTRAINT chk_supplier_contact_length CHECK (LENGTH(contact) = 10);

CREATE TABLE Branch(
   Branch_id int PRIMARY KEY,
```

```
    Address TEXT,
    Pincode char(6) NOT NULL,
    Provider_id int,
    FOREIGN KEY(Provider_id) REFERENCES Supplier(Supplier_id)
);

INSERT INTO Branch(Branch_id, Address, Pincode, Provider_id)
VALUES
(1,'A-144, Lajpat Nagar','110024',5),
(2,'Z-12, Karol Bagh','110025',2),
(3,'RW-42, Janakpuri','110026',3),
(4,'F-3, Hauz Khas','110027',8),
(5,'A-23, Najafgarh','110028',10),
(6,'G-107, Okhla','110030',9),
(7,'H-27, Paschim Vihar','110029',7),
(8,'G-1/56, Uttam Nagar','110059',8),
(9,'G-2/45, Vikas Puri','110058',6),
(10,'F-46, Nangloi','110061',1);

ALTER TABLE Branch
ADD CONSTRAINT chk_branch_pincode_length CHECK (LENGTH(Pincode) = 6);

ALTER TABLE Branch
ADD CONSTRAINT fk_branch_provider
FOREIGN KEY (Provider_id) REFERENCES Supplier(Supplier_id);

CREATE TABLE Employee(
    Employee_id varchar(10) PRIMARY KEY,
    name varchar(255),
    contact char(10) UNIQUE,
    Branch_id int,
```

```
   FOREIGN KEY (Branch_id) REFERENCES Branch(Branch_id)
);

ALTER TABLE Employee
ADD CONSTRAINT chk_employee_id_prefix CHECK (Employee_id LIKE 'abcdx%');

-- ALTER TABLE Employee
-- ADD CONSTRAINT chk_employee_contact_length CHECK (LENGTH(contact) = 10);

INSERT INTO Employee(Employee_id, name, contact, Branch_id)
VALUES
('abcdx1', 'John Doe', '9876543210', 5),
('abcdx2', 'Jane Smith', '1234567829', 2),
('abcdx3', 'Alice Johnson', '9876543221', 3),
('abcdx4', 'Bob Williams', '1234567899', 8),
('abcdx5', 'Mary Brown', '9876543218', 10),
('abcdx6', 'David Miller', '1234567897', 9),
('abcdx7', 'Emily Wilson', '9876543215', 7),
('abcdx8', 'Michael Davis', '1234567894', 8),
('abcdx9', 'Olivia Garcia', '9876543212', 6),
('abcdx10', 'James Rodriguez', '1234567893', 1);

-- Creating Inventory table
-- @block
CREATE TABLE Inventory(
Branch_id INT NOT NULL,
Item_id int ,
Quantity int,
Supplier_id int
);
-- @block
```

```
INSERT Into Inventory(Branch_id,Item_id,Quantity,Supplier_id)
values
(1,4001,850, 2),
(1,4002,130, 3),
(1,4003,450, 5),
(1,4004,630, 1),
(1,4005,710, 4),
(2,4001,230, 7),
(2,4002,250, 3),
(2,4003,150, 5),
(2,4004,300, 8),
(2,4005,500, 4);

-- Creating Item table
CREATE TABLE Item(
   Item_id INT PRIMARY KEY,
   Name varchar(100),
   Description TEXT,
   Price int
);

-- Adding constraint for Item_id
ALTER TABLE Item
ADD CONSTRAINT chk_item_id_prefix CHECK (Item_id >= 4000);

-- Inserting data into Item table
INSERT INTO Item(Item_id, Name, Description, Price)
VALUES
(4001, 'Horlicks', 'Health Drink', 70),
(4002, 'Colgate', 'Toothpaste', 50),
(4003, 'Dove', 'Shampoo', 120),
```

```
(4004, 'KitKat', 'Chocolate Bar', 30),
(4005, 'Lays', 'Potato Chips', 20),
(4006, 'Tide', 'Laundry Detergent', 80),
(4007, 'Coca-Cola', 'Soft Drink', 40),
(4008, 'Nivea', 'Body Lotion', 100),
(4009, 'Pringles', 'Potato Crisps', 25),
(4010, 'Dettol', 'Antiseptic Liquid', 60);

-- Creating User table
CREATE TABLE User(
User_id varchar(100) PRIMARY KEY,
name varchar(255),
contact char(10) NOT NULL UNIQUE,
Address TEXT NOT NULL,
Pincode char(6) NOT NULL,
CHECK (User_id LIKE 'U1%')
);

-- @block
CREATE INDEX idx_User_User_id ON User (User_id);
-- @block
INSERT INTO User (User_id, name, contact, Address, Pincode)
VALUES
('U1001', 'John Doe', '9876543210', 'A-144, Lajpat Nagar', '110024'),
('U1002', 'Jane Smith', '1234567890', 'Z-12, Karol Bagh', '110025'),
('U1003', 'Alice Johnson', '9876543211', 'RW-42, Janakpuri', '110026'),
('U1004', 'Bob Williams', '1234567891', 'F-3, Hauz Khas', '110027'),
('U1005', 'Mary Brown', '9876543212', 'A-23, Najafgarh', '110028'),
('U1006', 'David Miller', '1234567892', 'G-107, Okhla', '110030'),
('U1007', 'Emily Wilson', '9876543213', 'H-27, Paschim Vihar', '110029'),
('U1008', 'Michael Davis', '1234567893', 'G-1/56, Uttam Nagar', '110059'),
```

```
('U1009', 'Olivia Garcia', '9876543214', 'G-2/45, Vikas Puri', '110058'),
('U1010', 'James Rodriguez', '1234567894', 'F-46, Nangloi', '110062');
-- @block
CREATE TABLE Order_table(
Order_id varchar(10),
User_id varchar(100) NOT NULL,
Branch_id int,
order_date Date,
status varchar(20),
PRIMARY KEY (Order_id),
FOREIGN KEY (User_id) REFERENCES User(User_id),
CHECK (Order_id LIKE 'hjgx%')
);
-- @block
CREATE INDEX idx_Order_table_Order_id ON Order_table (Order_id);
CREATE INDEX idx_Order_table_User_id ON Order_table (User_id);
CREATE INDEX idx_Order_table_Branch_id ON Order_table (Branch_id);
-- @block
INSERT INTO Order_table(Order_id, User_id, Branch_id, order_date, status)
VALUES
('hjgx1', 'U1001', 001, '2024-02-12', 'on the way'),
('hjgx2', 'U1002', 002, '2024-02-12', 'Delivered'),
('hjgx3', 'U1003', 003, '2024-02-12', 'packing'),
('hjgx4', 'U1004', 004, '2024-02-12', 'on the way'),
('hjgx5', 'U1005', 005, '2024-02-12', 'Delivered'),
('hjgx6', 'U1006', 006, '2024-02-12', 'packing'),
('hjgx7', 'U1007', 007, '2024-02-12', 'on the way'),
('hjgx8', 'U1008', 008, '2024-02-12', 'Delivered'),
('hjgx9', 'U1009', 009, '2024-02-12', 'packing'),
('hjgx10', 'U1010', 0010, '2024-02-12', 'on the way');
```

```
-- Creating Transaction table
CREATE TABLE Transaction(
Transaction_id varchar(10) PRIMARY KEY,
Order_id varchar(10) NOT NULL,
Item_id int,
Quantity int,
FOREIGN KEY (Order_id) REFERENCES Order_table(Order_id),
CHECK (Transaction_id LIKE 'T%')
);

-- Adding constraint for Transaction_id
ALTER TABLE Transaction
ADD CONSTRAINT chk_transaction_id_prefix CHECK (Transaction_id LIKE 'T%');

-- Inserting data into Transaction table
INSERT INTO Transaction(Transaction_id, Order_id, Item_id, Quantity)
VALUES
('T001', 'hjgx1', 4001, 2),
('T002', 'hjgx1', 4003, 1),
('T003', 'hjgx1', 4005, 3),
('T004', 'hjgx1', 4002, 2),
('T005', 'hjgx1', 4007, 1),
('T006', 'hjgx2', 4004, 2),
('T007', 'hjgx2', 4006, 1),
('T008', 'hjgx2', 4008, 3),
('T009', 'hjgx2', 4009, 2),
('T010', 'hjgx2', 4010, 1);
```

```
-- @block
CREATE TABLE Feedback (
ID INT AUTO_INCREMENT PRIMARY KEY,
OrderID varchar(10) NOT NULL,
feedback_text TEXT,
rating INT NOT NULL,
Userid varchar(100),
FOREIGN KEY (OrderID) REFERENCES Order_table(Order_id),
FOREIGN KEY (Userid) REFERENCES User(User_id)
);
-- Add an index for the 'ID' column
CREATE INDEX idx_Feedback_ID ON Feedback(ID);
-- @block
CREATE INDEX idx_Feedback_OrderID ON Feedback (OrderID);
CREATE INDEX idx_Feedback_Userid ON Feedback (Userid);
-- @block
INSERT INTO Feedback (OrderID, feedback_text, rating, Userid)
VALUES
('hjgx1', 'Great service and fast delivery!', 5, 'U1001'),
('hjgx2', 'The product quality is excellent.', 4, 'U1002'),
('hjgx3', 'Average experience, could be better.', 3, 'U1003'),
('hjgx4', 'Very satisfied with the purchase.', 5, 'U1004'),
('hjgx5', 'Delivery was delayed, but the product is good.', 4, 'U1005'),
('hjgx6', 'The customer support was helpful.', 5, 'U1006'),
('hjgx7', 'Product was damaged during transit.', 2, 'U1007'),
('hjgx8', 'The item received did not match the description.', 2, 'U1008'),
('hjgx9', 'Prompt delivery and good packaging.', 4, 'U1009'),
('hjgx10', 'The product is exactly what I wanted.', 5, 'U1010');
-- @block
CREATE TABLE Employee_Branch (
Employee_id varchar(10),
```

```
Branch_id int,
FOREIGN KEY (Employee_id) REFERENCES Employee(Employee_id),
FOREIGN KEY (Branch_id) REFERENCES Branch(Branch_id),
PRIMARY KEY (Employee_id, Branch_id)
);
-- @block
CREATE TABLE Inventory_Item (
Branch_id INT,
Item_id INT,
Quantity INT,
Supplier_id INT,
FOREIGN KEY (Branch_id) REFERENCES Branch(Branch_id),
FOREIGN KEY (Item_id) REFERENCES Item(Item_id),
FOREIGN KEY (Supplier_id) REFERENCES Supplier(Supplier_id),
PRIMARY KEY (Branch_id, Item_id)
);
-- @block
CREATE TABLE Transaction_Item (
Transaction_id varchar(10),
Item_id INT,
Quantity INT,
FOREIGN KEY (Transaction_id) REFERENCES Transaction(Transaction_id),
FOREIGN KEY (Item_id) REFERENCES Item(Item_id),
PRIMARY KEY (Transaction_id, Item_id)
);
-- @block
CREATE TABLE Order_Feedback (
OrderID varchar(10),
FeedbackID int,
FOREIGN KEY (OrderID) REFERENCES Order_table(Order_id),
FOREIGN KEY (FeedbackID) REFERENCES Feedback(ID),
```

```
PRIMARY KEY (OrderID, FeedbackID)
);

-- @block
ALTER TABLE Order_table
ADD CONSTRAINT fk_order_user
FOREIGN KEY (User_id) REFERENCES User(User_id);



-- Inserting data into Employee_Branch table
INSERT INTO Employee_Branch (Employee_id, Branch_id)
VALUES
('abcdx1', 5),
('abcdx2', 2),
('abcdx3', 3),
('abcdx4', 8),
('abcdx5', 10),
('abcdx6', 9),
('abcdx7', 7),
('abcdx8', 8),
('abcdx9', 6),
('abcdx10', 1);

-- Inserting data into Inventory_Item table
INSERT INTO Inventory_Item (Branch_id, Item_id, Quantity, Supplier_id)
VALUES
(1, 4001, 850, 2),
(1, 4002, 130, 3),
(1, 4003, 450, 5),
(1, 4004, 630, 1),
(1, 4005, 710, 4),
```

```
(2, 4001, 230, 7),
(2, 4002, 250, 3),
(2, 4003, 150, 5),
(2, 4004, 300, 8),
(2, 4005, 500, 4);

-- Inserting data into Transaction_Item table
INSERT INTO Transaction_Item (Transaction_id, Item_id, Quantity)
VALUES
('T001', 4001, 2),
('T001', 4003, 1),
('T001', 4005, 3),
('T001', 4002, 2),
('T001', 4007, 1),
('T002', 4004, 2),
('T002', 4006, 1),
('T002', 4008, 3),
('T002', 4009, 2),
('T002', 4010, 1);

-- @block
-- DROP TABLE feedback;
-- --@block
-- DROP TABLE IF EXISTS Transaction_Item;
-- DROP TABLE IF EXISTS Inventory_Item;
-- DROP TABLE IF EXISTS Employee_Branch;
```

```
-- DROP TABLE IF EXISTS Feedback;
-- DROP TABLE IF EXISTS Order_table;
-- DROP TABLE IF EXISTS User;
-- DROP TABLE IF EXISTS Transaction;
-- DROP TABLE IF EXISTS Item;
-- DROP TABLE IF EXISTS Inventory;
-- DROP TABLE IF EXISTS Employee;
-- DROP TABLE IF EXISTS Branch;
-- DROP TABLE IF EXISTS Supplier;


-- 10 Queries

-- Select all users with their addresses and contact numbers:
SELECT User_id, name, Address, contact
FROM User;

-- Update the status of order 'hjgx2' to 'Shipped':
UPDATE Order_table
SET status = 'Shipped'
WHERE Order_id = 'hjgx2';

-- Find the total quantity of each item in the inventory along with their descriptions:
SELECT i.Item_id, i.Name AS Item_Name, i.Description, SUM(Quantity) AS Total_Quantity
FROM Inventory inv
JOIN Item i ON inv.Item_id = i.Item_id
GROUP BY i.Item_id;


-- Insert a new user with the User_id 'U1011', name 'Sarah Jones', contact '9876543211', address 'B-12, Saket', and pincode '110017':
```

```
INSERT INTO User (User_id, name, contact, Address, Pincode)
VALUES ('U1011', 'Sarah Jones', '9876545211', 'B-12, Saket', '110017');

-- Select the details of the items with their corresponding suppliers:
SELECT i.Item_id, i.Name AS Item_Name, s.name AS Supplier_Name
FROM Item i
JOIN Inventory inv ON i.Item_id = inv.Item_id
JOIN Supplier s ON inv.Supplier_id = s.Supplier_id;

-- Display the orders along with their respective user names and addresses:
SELECT o.Order_id, u.name AS User_Name, u.Address AS User_Address
FROM Order_table o
JOIN User u ON o.User_id = u.User_id;

-- Retrieve the orders placed by users whose names start with 'J':
SELECT o.Order_id, o.User_id, u.name, o.order_date, o.status
FROM Order_table o
JOIN User u ON o.User_id = u.User_id
WHERE u.name LIKE 'J%';

-- Update the contact number of the user with User_id 'U1003' to '9876543212':
UPDATE User
SET contact = '9886543212'
WHERE User_id = 'U1003';

-- Calculate the total price for each order considering the quantity and price of each item:
SELECT o.Order_id, SUM(i.Price * t.Quantity) AS Total_Price
FROM Transaction t
JOIN Item i ON t.Item_id = i.Item_id
JOIN Order_table o ON t.Order_id = o.Order_id
GROUP BY o.Order_id;
```

```
-- Display the information of employees working in the branch with Branch_id = 005:
select * from employee where branch_id = 5;




-- Constraint (phone number)
-- UPDATE User
-- SET contact = '9876543212'
-- WHERE User_id = 'U1003';

CREATE TABLE login_attempt (
    id INT AUTO_INCREMENT PRIMARY KEY,
    User_id VARCHAR(100),
    attempt_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);




-- Triggers

DELIMITER //

CREATE TRIGGER before_transaction_branch_check
BEFORE INSERT ON Transaction
FOR EACH ROW
BEGIN
    DECLARE branch_id_found INT;
    DECLARE user_id_inserted VARCHAR(100);

    -- Get the Order_id of the current transaction
```

```
    SET user_id_inserted = (SELECT User_id FROM Order_table WHERE Order_id = NEW.Order_id);

    -- Get the Branch_id based on the User's pincode
    SELECT Branch_id INTO branch_id_found FROM Branch WHERE Pincode = (SELECT Pincode FROM
User WHERE User_id = user_id_inserted);

    -- If branch not found, raise an error
    IF branch_id_found IS NULL THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Store is not available in your region.';
    END IF;
END//

DELIMITER ;



DELIMITER //

CREATE TRIGGER before_transaction_item_check
BEFORE INSERT ON Transaction
FOR EACH ROW
BEGIN
    DECLARE branch_id_found INT;
    DECLARE item_id_inserted INT;
    DECLARE user_id_inserted VARCHAR(100);

    -- Get the Item_id being inserted
    SET item_id_inserted = NEW.Item_id;

    -- Get the Order_id of the current transaction
    SET user_id_inserted = (SELECT User_id FROM Order_table WHERE Order_id = NEW.Order_id);
```

```
    -- Get the Branch_id based on the User's pincode
    SELECT Branch_id INTO branch_id_found FROM Branch WHERE Pincode = (SELECT Pincode FROM User WHERE User_id = user_id_inserted);

    -- Check item existence and quantity in inventory
    IF NOT EXISTS (SELECT * FROM Inventory WHERE Branch_id = branch_id_found AND Item_id = item_id_inserted AND Quantity >= NEW.Quantity) THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Item is out of stock.';
    END IF;
END//

DELIMITER ;

DELIMITER //

CREATE TRIGGER before_login_attempt
BEFORE INSERT ON Login_attempt
FOR EACH ROW
BEGIN
    DECLARE user_exists INT;

    -- Check if the user exists
    SELECT COUNT(*) INTO user_exists FROM User WHERE User_id = NEW.User_id;

    -- If user does not exist, raise an error
    IF user_exists = 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'User does not exist.';
    END IF;
END//

DELIMITER ;
```

```
START TRANSACTION;
INSERT INTO User (User_id, name, contact, Address, Pincode)
VALUES ('U1012', 'Emily Watson', '9876543222', 'C-34, Rohini', '110085');
COMMIT;

START TRANSACTION;
UPDATE User
SET contact = '9876545222'
WHERE User_id = 'U1003';
COMMIT;

START TRANSACTION;
INSERT INTO Order_table (Order_id, User_id, Branch_id, order_date, status)
VALUES ('hjgx11', 'U1012', 6, '2024-04-20', 'processing');
COMMIT;

START TRANSACTION;
INSERT INTO Feedback (OrderID, feedback_text, rating, Userid)
VALUES ('hjgx11', 'Good service!', 4, 'U1012');
COMMIT;

-- START TRANSACTION;
-- UPDATE User
-- SET contact = '9876545223'
-- WHERE User_id = 'U1003';
-- ROLLBACK;


START TRANSACTION;
```

```
INSERT INTO Order_table (Order_id, User_id, Branch_id, order_date, status)
VALUES ('hjgx12', 'U1012', 6, '2024-04-20', 'processing');
COMMIT;

INSERT INTO Order_table (Order_id, User_id, Branch_id, order_date, status)
VALUES ('hjgx13', 'U1012', 6, '2024-04-20', 'processing');
COMMIT;

-- START TRANSACTION;
-- UPDATE Order_table
-- SET status = 'Shipped'
-- WHERE Order_id = 'hjgx11';
-- COMMIT;

-- START TRANSACTION;
-- INSERT INTO Order_table (Order_id, User_id, Branch_id, order_date, status)
-- VALUES ('hjgx11', 'U1006', 6, '2024-04-20', 'processing');
-- COMMIT;



-- START TRANSACTION;
-- INSERT INTO Feedback (OrderID, feedback_text, rating, Userid)
-- VALUES ('hjgx999', 'This order was great!', 5, 'U1005');
-- COMMIT;
```