# **SQL QUERIES (DDL)**

1. **USERS TABLE**

CREATE TABLE USERS(

user\_name VARCHAR2(30) PRIMARY KEY,

name VARCHAR2(30) NOT NULL,

user\_role VARCHAR2(10) DEFAULT 'Cashier' NOT NULL,

user\_password VARCHAR2(15) NOT NULL

);

ALTER TABLE users

ADD CHECK (LENGTH(user\_password) >= 8);

1. **RECEIPT TABLE**

CREATE TABLE receipt(

rec\_no INT PRIMARY KEY,

rec\_date DATE NOT NULL,

amount NUMBER DEFAULT 0 NOT NULL,

tax NUMBER DEFAULT 0 NOT NULL,

total\_amount NUMBER DEFAULT 0 NOT NULL,

refund\_issued INT DEFAULT 0 NOT NULL

);

ALTER TABLE receipt

ADD user\_name VARCHAR2(30) NOT NULL;

ALTER TABLE receipt

ADD CONSTRAINT receipt\_rec\_no\_fk FOREIGN KEY(user\_name) REFERENCES users(user\_name)

ON DELETE CASCADE;

ALTER TABLE receipt

ADD CHECK (refund\_issued IN (0, 1));

CREATE SEQUENCE rec\_no\_SEQ

START WITH 1

INCREMENT BY 1

NOCACHE;

CREATE OR REPLACE TRIGGER receipt\_insert\_trg

BEFORE INSERT ON receipt

FOR EACH ROW

BEGIN

:NEW.rec\_no:= rec\_no\_SEQ.NEXTVAL;

END;

ALTER TABLE receipt

ADD sales\_id INT NOT NULL;

ALTER TABLE receipt

ADD CONSTRAINT receipt\_sale\_fk FOREIGN KEY(sales\_id) REFERENCES sales(sales\_id)

ON DELETE CASCADE;

ALTER TABLE receipt

MODIFY sales\_id NULL;

1. **CUSTOMER TABLE**

CREATE TABLE customer(

cus\_cnic VARCHAR2(15) PRIMARY KEY,

cus\_name VARCHAR2(30) NOT NULL,

cus\_address VARCHAR2(30),

cus\_contact VARCHAR(12) NOT NULL

);

ALTER TABLE customer

ADD CHECK (LENGTH(CUS\_CNIC) = 15 AND

SUBSTR(CUS\_CNIC, 1, 5) BETWEEN '00000' AND '99999' AND

SUBSTR(CUS\_CNIC, 6, 1) = '-' AND

SUBSTR(CUS\_CNIC, 7, 7) BETWEEN '0000000' AND '9999999' AND

SUBSTR(CUS\_CNIC, 14, 1) = '-' AND

SUBSTR(CUS\_CNIC, 15, 1) BETWEEN '0' AND '9');

ALTER TABLE customer

ADD CHECK (SUBSTR(cus\_contact,1,4) BETWEEN '0000' AND '9999'

AND SUBSTR(cus\_contact, 5, 1) = '-'

AND SUBSTR(cus\_contact,6) BETWEEN '0000000' AND '9999999');

ALTER TABLE customer

DROP COLUMN cus\_address

ALTER TABLE customer

ADD city\_code VARCHAR(5) NOT NULL

ALTER TABLE customer

ADD CONSTRAINT cus\_city\_code\_fk FOREIGN KEY(city\_code) REFERENCES address(city\_code);

1. **VENDOR TABLE**

CREATE TABLE vendor(

ven\_id INT PRIMARY KEY,

ven\_name VARCHAR2(30) NOT NULL,

ven\_address VARCHAR2(30),

ven\_contact VARCHAR(2) NOT NULL

);

ALTER TABLE vendor

ADD CHECK (SUBSTR(ven\_contact,1,4) BETWEEN '0000' AND '9999'

AND SUBSTR(ven\_contact, 5, 1) = '-'

AND SUBSTR(ven\_contact,6) BETWEEN '0000000' AND '9999999');

ALTER TABLE vendor

DROP COLUMN ven\_address

ALTER TABLE vendor

ADD city\_code VARCHAR(5) NOT NULL

ALTER TABLE vendor

ADD CONSTRAINT ven\_city\_code\_fk FOREIGN KEY(city\_code) REFERENCES address(city\_code);

ALTER TABLE vendor

MODIFY ven\_contact VARCHAR2(12);

1. **PURCHASES TABLE**

CREATE TABLE purchases(

pur\_id INT PRIMARY KEY,

pur\_date DATE NOT NULL,

pur\_amount NUMBER NOT NULL,

ven\_id INT NOT NULL

);

ALTER TABLE purchases

ADD CONSTRAINT pur\_ven\_id\_fk FOREIGN KEY(ven\_id) REFERENCES vendor(ven\_id)

ON DELETE CASCADE;

CREATE SEQUENCE PUR\_ID\_SEQ

START WITH 1

INCREMENT BY 1

NOCACHE;

CREATE OR REPLACE TRIGGER purchases\_insert\_trg

BEFORE INSERT ON purchases

FOR EACH ROW

BEGIN

:NEW.pur\_id:= pur\_id\_SEQ.NEXTVAL;

END;

1. **EXPENSES TABLE**

CREATE TABLE expenses(

exp\_id INT PRIMARY KEY,

exp\_amount NUMBER NOT NULL,

exp\_type VARCHAR2(20) NOT NULL,

exp\_name VARCHAR2(30) NOT NULL,

exp\_date DATE NOT NULL,

ven\_id INT NOT NULL

);

CREATE SEQUENCE EXP\_ID\_SEQ

START WITH 1

INCREMENT BY 1

NOCACHE;

CREATE OR REPLACE TRIGGER expenses\_insert\_trg

BEFORE INSERT ON expenses

FOR EACH ROW

BEGIN

:NEW.exp\_id:= exp\_id\_SEQ.NEXTVAL;

END;

ALTER TABLE expenses

ADD CONSTRAINT exp\_ven\_id\_fk FOREIGN KEY(ven\_id) REFERENCES vendor(ven\_id)

ON DELETE CASCADE;

1. **RACK TABLE**

CREATE TABLE RACK(

rack\_no INT PRIMARY KEY,

rack\_location VARCHAR2(15) NOT NULL,

rack\_capacity INT DEFAULT 50 NOT NULL,

rack\_strength INT DEFAULT 0 NOT NULL

);

ALTER TABLE rack

ADD cat\_name VARCHAR2(30) NOT NULL;

ALTER TABLE rack

ADD CHECK (rack\_strength <= rack\_capacity);

ALTER TABLE rack

ADD CONSTRAINT cat\_name\_fk FOREIGN KEY(cat\_name) REFERENCES category(cat\_name)

ON DELETE CASCADE;

1. **CATEGORY TABLE**

CREATE TABLE category(

cat\_name VARCHAR2(30) PRIMARY KEY,

cat\_description VARCHAR2(30) NOT NULL

);

ALTER TABLE category

MODIFY cat\_description VARCHAR2(100);

1. **ITEM TABLE**

CREATE TABLE item(

item\_id INT PRIMARY KEY,

item\_name VARCHAR2(30) NOT NULL,

item\_quantity INT DEFAULT 0 NOT NULL,

item\_price NUMBER NOT NULL,

reorder\_point INT NOT NULL,

item\_desc VARCHAR2(30),

item\_barcode INT UNIQUE NOT NULL,

cat\_name VARCHAR2(30) NOT NULL,

rack\_no INT NOT NULL

);

ALTER TABLE item

ADD CONSTRAINT item\_cat\_name\_fk FOREIGN KEY(cat\_name) REFERENCES category(cat\_name)

ON DELETE CASCADE;

ALTER TABLE item

ADD CONSTRAINT item\_rack\_no\_fk FOREIGN KEY(rack\_no) REFERENCES rack(rack\_no)

ON DELETE CASCADE;

CREATE SEQUENCE ITEM\_ID\_SEQ

START WITH 1

INCREMENT BY 1

NOCACHE;

CREATE OR REPLACE TRIGGER item\_insert\_trg

BEFORE INSERT ON item

FOR EACH ROW

BEGIN

:NEW.item\_id:= item\_id\_SEQ.NEXTVAL;

END;

ALTER TABLE item

MODIFY item\_desc VARCHAR2(100);

1. **PURCHASE\_ITEMS TABLE**

CREATE TABLE purchase\_items(

item\_id INT NOT NULL,

pur\_id INT NOT NULL,

unit\_price NUMBER NOT NULL,

pur\_quantity INT NOT NULL,

total\_amount NUMBER NOT NULL

);

ALTER TABLE purchase\_items

DROP COLUMN total\_amount;

ALTER TABLE purchase\_items

ADD CONSTRAINT purchase\_items\_pk PRIMARY KEY(item\_id, pur\_id);

ALTER TABLE purchase\_items

ADD CONSTRAINT purchase\_items\_\_item\_fk FOREIGN KEY(item\_id) REFERENCES item(item\_id)

ON DELETE CASCADE;

ALTER TABLE purchase\_items

ADD CONSTRAINT purchase\_items\_\_pur\_fk FOREIGN KEY(pur\_id) REFERENCES purchases(pur\_id)

ON DELETE CASCADE;

1. **SALES TABLE**

CREATE TABLE sales(

sales\_id INT PRIMARY KEY,

sales\_date DATE NOT NULL,

sales\_amount NUMBER NOT NULL,

cus\_cnic VARCHAR2(15) NOT NULL

);

CREATE SEQUENCE SALES\_ID\_SEQ

START WITH 1

INCREMENT BY 1

NOCACHE;

CREATE OR REPLACE TRIGGER sales\_insert\_trg

BEFORE INSERT ON sales

FOR EACH ROW

BEGIN

:NEW.sales\_id:= sales\_id\_SEQ.NEXTVAL;

END;

ALTER TABLE sales

ADD CONSTRAINT sales\_fk FOREIGN KEY(cus\_cnic) REFERENCES customer(cus\_cnic)

ON DELETE CASCADE;

1. **SALES\_ITEMS TABLE**

CREATE TABLE sales\_items(

sales\_id INT NOT NULL,

item\_id INT NOT NULL,

sale\_quantity INT NOT NULL,

unit\_price NUMBER NOT NULL

);

ALTER TABLE sales\_items

ADD CONSTRAINT sales\_items\_pk PRIMARY KEY(sales\_id, item\_id);

ALTER TABLE sales\_items

ADD CONSTRAINT sales\_items\_sale\_fk FOREIGN KEY(sales\_id) REFERENCES sales(sales\_id)

ON DELETE CASCADE;

ALTER TABLE sales\_items

ADD CONSTRAINT sales\_items\_item\_fk FOREIGN KEY(item\_id) REFERENCES item(item\_id)

ON DELETE CASCADE;

1. **SALES\_ITEMS TABLE**

CREATE TABLE address(

city\_code VARCHAR(5) PRIMARY KEY,

city VARCHAR2(15) NOT NULL,

country VARCHAR2(15) NOT NULL

);

# **SQL QUERIES (DML)**

INSERT INTO users

VALUES('ali.mustafa', 'Muhammad Ali Mustafa', 'Admin', '57365736');

# **SQL QUERIES (PL/SQL)**

CREATE OR REPLACE FUNCTION check\_username (username USERS.USER\_NAME%TYPE)

RETURN USERS.USER\_PASSWORD%TYPE

IS

pass USERS.USER\_PASSWORD%TYPE;

BEGIN

SELECT USER\_password INTO pass

FROM users

WHERE user\_name = username;

RETURN pass;

END;

CREATE OR REPLACE TRIGGER updateItemsCategory

AFTER UPDATE OF cat\_name ON rack

FOR EACH ROW

BEGIN

UPDATE item SET cat\_name = :NEW.cat\_name WHERE cat\_name = :OLD.cat\_name;

END;

CREATE OR REPLACE TRIGGER updateRackStrength

AFTER INSERT ON item

FOR EACH ROW

BEGIN

UPDATE rack SET rack\_strength = rack\_strength + :NEW.item\_quantity

WHERE rack\_no = :NEW.rack\_no;

END;

CREATE OR REPLACE TRIGGER updateRackStrengthOnDelete

AFTER DELETE ON item

FOR EACH ROW

BEGIN

UPDATE rack SET rack\_strength = rack\_strength - :OLD.item\_quantity

WHERE rack\_no = :OLD.rack\_no;

END;

CREATE OR REPLACE TRIGGER updateRackStrengthOnUpdate

AFTER UPDATE OF item\_quantity ON item

FOR EACH ROW

BEGIN

UPDATE rack SET rack\_strength = rack\_strength - :OLD.item\_quantity

WHERE rack\_no = :OLD.rack\_no;

UPDATE rack SET rack\_strength = rack\_strength + :NEW.item\_quantity

WHERE rack\_no = :OLD.rack\_no;

END;

CREATE OR REPLACE PROCEDURE salesRecordDelete (recNo NUMBER)

IS

salesID NUMBER;

BEGIN

SELECT sales\_id INTO salesID

FROM receipt

WHERE rec\_no = recNo;

DELETE FROM receipt

WHERE rec\_no = recNo;

DELETE FROM sales\_items

WHERE sales\_id = salesID;

DELETE FROM sales

WHERE sales\_id = salesID;

END;