**1)**

i)

CREATE TABLE Article (

ArCode CHAR(5) PRIMARY KEY CHECK (ArCode LIKE 'A%'),

ArName VARCHAR2(30) NOT NULL,

Rate NUMBER(8,2),

Quantity NUMBER(4) DEFAULT 0 CHECK (Quantity >= 0),

Class CHAR(1) CHECK (Class IN ('A', 'B', 'C'))

);

ii)

INSERT INTO Article (ArCode, ArName, Rate, Quantity, Class)

VALUES ('A1001', 'Mouse', 500, 0, 'C');

iii)

ALTER TABLE Article

MODIFY ArCode CHAR(10);

iv)

ALTER TABLE Article

ADD ArISBM VARCHAR(20);

UPDATE Article

SET ArISBM = 'New\_Value'

WHERE ArCode = 'A1001';

v)

ALTER TABLE Article

MODIFY Quantity NUMBER(4);

vi)

ALTER TABLE Article

RENAME COLUMN rate TO price;

**2)**

i)

SELECT Itemcode, Descr, Price

FROM Item

WHERE (Descr LIKE '%Shirt%' OR Descr LIKE '%Skirt%')

AND Category = 'B';

ii)

SELECT Itemcode, Descr, Price

FROM Item;

iii)

SELECT Quotationid, Sname

FROM Quotation

WHERE Qstatus IN ('Accepted', 'Rejected');

iv)

SELECT Itemcode, Descr, Price

FROM Item

WHERE SUBSTR(Descr, 2, 1) = 'r';

**3)**

i)

SELECT salesman\_id, name, city, commission

FROM Salesman

WHERE city IN ('Paris', 'Rome');

ii)

SELECT salesman\_id, name, city, commission

FROM Salesman

WHERE city NOT IN ('Paris', 'Rome');

iii)

SELECT salesman\_id, name, city, commission

FROM Salesman

WHERE commission BETWEEN 0.12 AND 0.14;

iv)

SELECT salesman\_id, name, city, commission

FROM Salesman

WHERE LEFT(name, 1) > 'A' AND LEFT(name, 1) < 'L';

v)

SELECT salesman\_id, name, city, commission

FROM Salesman

WHERE commission = (SELECT MAX(commission) FROM Salesman);

vi)

SELECT SUBSTR(name, INSTR(name, ' ') + 1) AS SecondHalfName

FROM Salesman;

**4)**

i)

SELECT Prodid, Pdesc, Category

FROM Product

WHERE Category = 'electronics';

ii)

SELECT Prodid, LEFT(Pdesc, 5) AS ShortDesc, Category

FROM Product;

iii)

SELECT COUNT(\*) AS TotalSales

FROM Sale;

v)

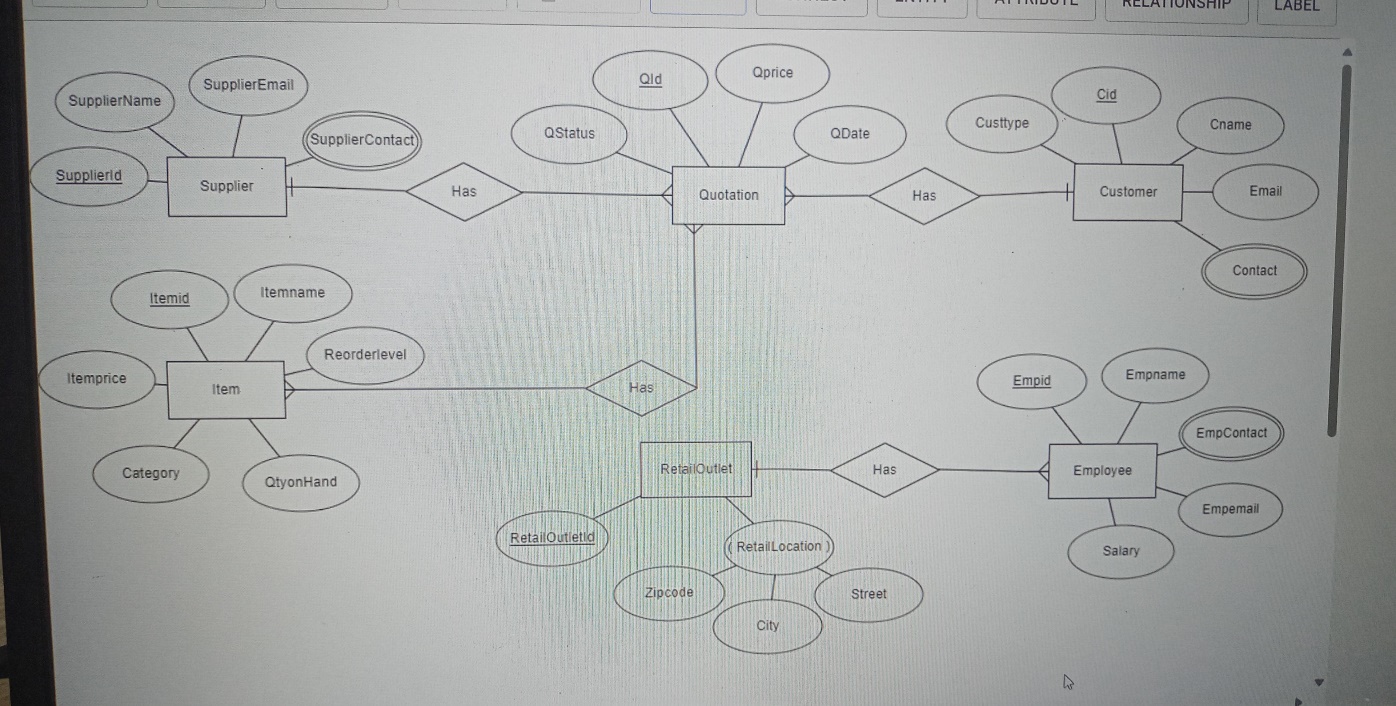
SELECT Prodid, Category, Discount

FROM Product

WHERE Category IN ('Sports', 'Apparel')

ORDER BY Category, Discount;

**5)**

****

i)

**Strong and Weak Entities:**

**Strong Entities:** These are entities that have their own existence and can be uniquely identified by their attributes. In this schema, entities like Supplier, Quotation, Customer, Item, RetailOutlet, and Employee are strong entities.

**Weak Entities:** These entities depend on another entity (usually a strong entity) for their existence. They don’t have a unique identifier on their own. In this schema, there are no obvious weak entities.

ii)

**Key Attributes:**

The key attributes are the ones that uniquely identify an entity. Let’s identify them for each entity:

Supplier: SupplierId

Quotation: QId

Customer: Cid

Item: Itemid

RetailOutlet: RetailOutletId

Employee: empid

iii)

**Degree of Relationships:**

The degree of a relationship refers to the number of entities involved in the relationship.

Let’s identify the degrees for each relationship:

\*Supplier has a relationship with Quotation (1-to-many): One supplier can have multiple quotations.

\*Customer has a relationship with Quotation (1-to-many): One customer can have multiple quotations.

\*Item has a relationship with Quotation (1-to-many): One item can be part of multiple quotations.

\*RetailOutlet has a relationship with Employee (1-to-many): One retail outlet can have multiple employees.

iv)

**Cardinality of Relationships:**

The cardinality describes the number of instances of one entity related to another entity.

Let’s mention the cardinalities for each relationship:

\*Supplier to Quotation: One supplier can be associated with multiple quotations (1-to-many).

\*Customer to Quotation: One customer can have multiple quotations (1-to-many).

\*Item to Quotation: One item can be part of multiple quotations (1-to-many).

\*RetailOutlet to Employee: One retail outlet can have multiple employees (1-to-many).

v)

**Key and Multi-Valued Attributes:**

Key attributes have already been identified above.

Multi-valued attributes are attributes that can have multiple values for a single entity. In this schema, there are no explicit multi-valued attributes.

**7)**

i)

SELECT \*

FROM Jobs\_Applied;

ii)

SELECT ApplicationId, JobseekerID, Jobpostid, jobTestResult

FROM Jobs\_Applied;

iii)

SELECT DISTINCT JobseekerID

FROM Jobs\_Applied;

iv)

SELECT \*

FROM Jobs\_Applied

WHERE jobTestResult = 'Open'

AND (jobAttemptStatus = 'No Attempt' OR jobAttemptStatus = 'Attempted');

v)

SELECT \*

FROM Jobs\_Applied

WHERE jobTestResult = 'Attempted'

AND jobSelectionStatus <> 'Selected';

**8)**

i)

SELECT Ename, Sal

FROM EMP

WHERE Sal > 2000;

ii)

SELECT Ename, Deptno

FROM EMP;

iii)

SELECT Empno, Ename, Job

FROM EMP

WHERE Comm IS NULL;

iv)

SELECT \*

FROM EMP

WHERE HireDate < '1981-09-01';

v)

SELECT \*

FROM EMP

WHERE SUBSTR(Ename, 2, 1) = 'I';

**9)**

a)

CREATE TABLE Books (

BookId VARCHAR(6) PRIMARY KEY,

Title VARCHAR(50) NOT NULL,

Author VARCHAR(20) NOT NULL,

Genre ENUM('Mystery', 'Thriller') NOT NULL,

PublicationYear INT

);

b)

ALTER TABLE Supplier

ADD city VARCHAR(10);

**10)**

a)

CREATE TABLE JOB (

JOB\_ID VARCHAR2(10) PRIMARY KEY,

JOB\_TITLE VARCHAR2(15) NOT NULL,

MIN\_SALARY NUMBER(6),

MAX\_SALARY NUMBER(6)

);

CREATE TABLE EMP (

EMP\_NUM CHAR(3) PRIMARY KEY CHECK (EMP\_NUM BETWEEN 1 AND 1000),

EMP\_LNAME VARCHAR2(15),

EMP\_FNAME VARCHAR2(15),

EMP\_INITIAL CHAR(1) CHECK (EMP\_INITIAL BETWEEN 'A' AND 'Z'),

EMP\_HIREDATE DATE NOT NULL,

JOB\_CODE VARCHAR2(10) REFERENCES JOB(JOB\_ID)

);

b)

ALTER TABLE JOB ADD STARS VARCHAR2(5);

c)

SELECT CONSTRAINT\_NAME, CONSTRAINT\_TYPE

FROM USER\_CONSTRAINTS

WHERE TABLE\_NAME IN ('EMP', 'JOB');

d)

DROP TABLE EMP CASCADE CONSTRAINTS;

DROP TABLE JOB CASCADE CONSTRAINTS;

**11)** CREATE TABLE Match (

Mid INTEGER PRIMARY KEY,

Tid INTEGER,

Player1 INTEGER,

Player2 INTEGER,

MatchDt DATE NOT NULL,

Winner INTEGER,

Score VARCHAR2(30) NOT NULL,

CHECK (Player1 <> Player2),

FOREIGN KEY (Tid) REFERENCES Tournament(Tid),

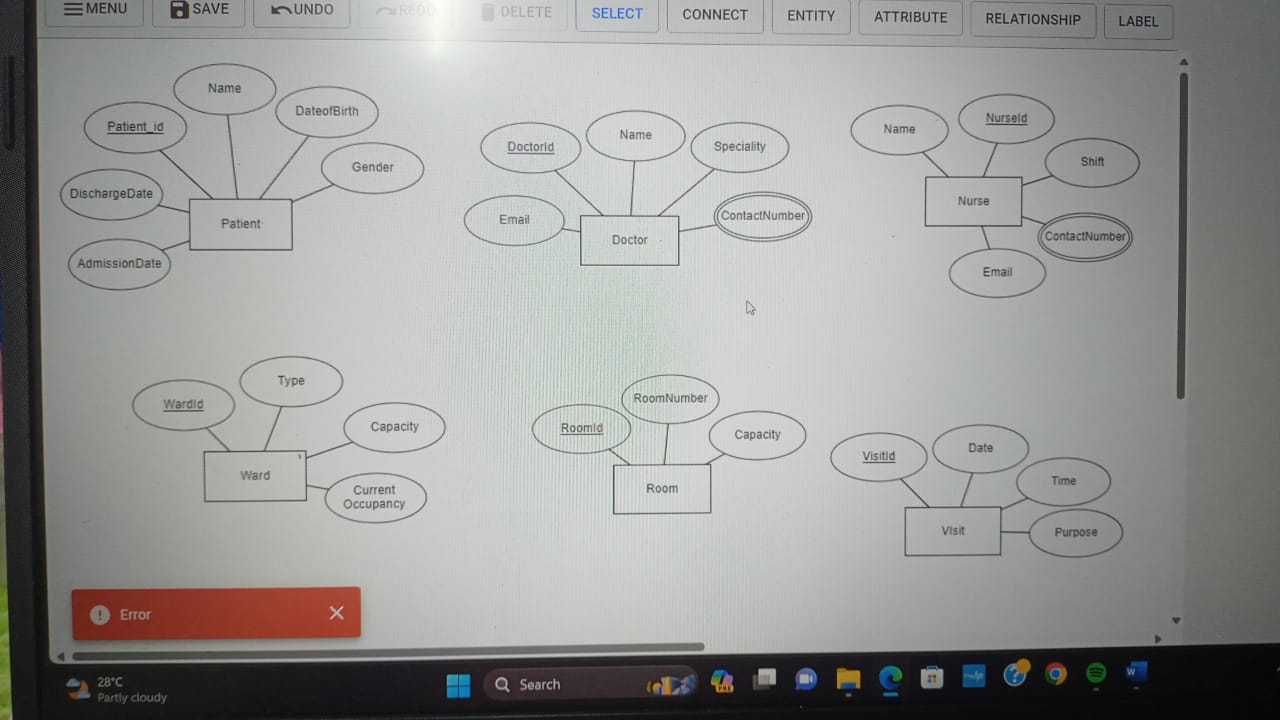
FOREIGN KEY (Player1) REFERENCES Player(Pid),

FOREIGN KEY (Player2) REFERENCES Player(Pid),

FOREIGN KEY (Winner) REFERENCES Player(Pid)

);

**12)**

****