Slot : A2 +TA2

School of Computer Science Engineering and Information Systems

Fall Semester 2024-2025

Continuous Assessment Test – I

Programme Name & Branch: MCA
Course Name & code: Database Systems & PMCA503L
Class Number (s): VL2024250103140, VL2024250103244, VL2024250103199
Faculty Name (s): Prof. Ranichandra C, Prof. Parimala M & Prof. Senthil Kumar N
Exam Duration: 90 Min. **Maximum Marks: 50**

KEY

1. XYZ Retail is a rapidly growing e-commerce company that manages a large and ever-expanding customer base. To maintain its competitive edge, the company relies on a centralized database that stores crucial information, including customer profiles, product inventories, sales invoices, payment records and customer feedback. Using the XYZ Retail scenario, answer the following questions and how the described changes affect the database's logical and physical data independence.

a) The company decides to add a new attribute, customer_city to the Customer entity to better analyse customer demographics. How does this change test the logical data independence of the system? What challenges might developers face and how can logical data independence help mitigate these challenges?

b) The company's database has grown significantly and the DBA decides to move some of the tables to a more efficient storage system to improve performance. How does this change test the physical data independence of the system? How should the system handle such a change without affecting the developers and end-users?

Solution

a) Adding a New Attribute: customer_city
Logical Data Independence:

- Logical data independence refers to the capacity to change the conceptual schema (i.e., the logical structure of the database, such as adding or modifying attributes in entities) without altering the external schemas or application programs. In other words, it means that changes at the logical level (like adding new fields or modifying tables) should not affect how the database is interacted with by applications or users.
- When XYZ Retail decides to add the customer_city attribute to the Customer entity, it changes the logical schema of the database. This modification tests logical data independence because it introduces a new attribute that applications may need to account for.
- Backward Compatibility**:** If the applications or user interfaces were designed without anticipating this new attribute, they might break or behave unexpectedly.
- **How Logical Data Independence Helps:** Logical data independence ensures that while the underlying structure of the database is updated (such as adding the customer_city attribute), the application programs should be able to function with minimal modification. Ideally, the database

design should be such that applications access data through a layer of abstraction, so changes to the database schema are managed with minimal disruption.

b) Moving Tables to a New Storage System
Physical Data Independence:

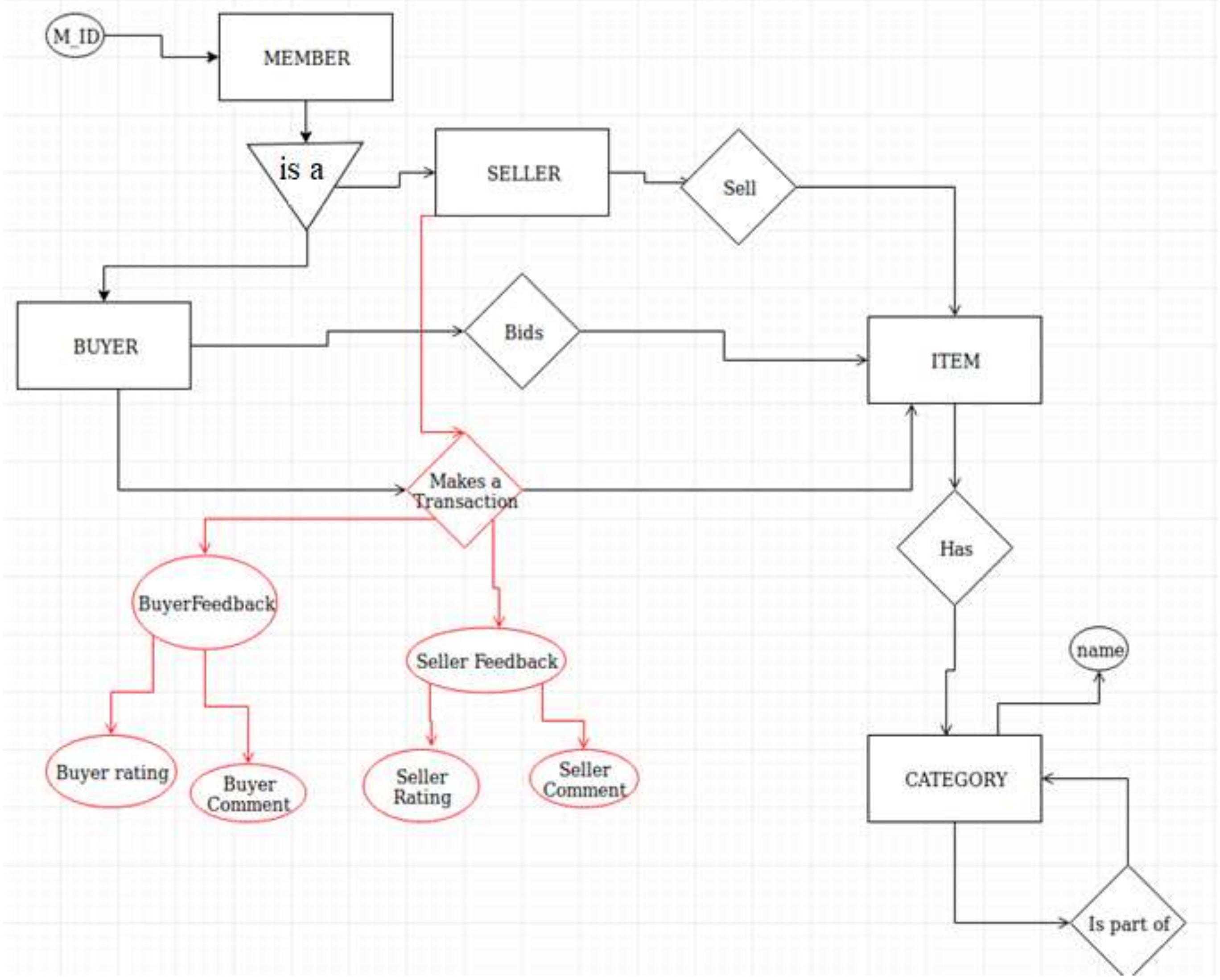
- Physical data independence refers to the capacity to change the physical storage of data without affecting the conceptual schema or the application programs. This means that changes in how and where data is physically stored (like moving tables to a new storage system) should not impact the logical structure or how the data is accessed and used by applications.
- **Impact of Moving Tables:** Moving tables to a more efficient storage system is a change at the physical level. It tests physical data independence by requiring that the database system's physical storage mechanisms can be modified without affecting the logical schema or the application access patterns.

2. Draw an EER diagram for the following situation.

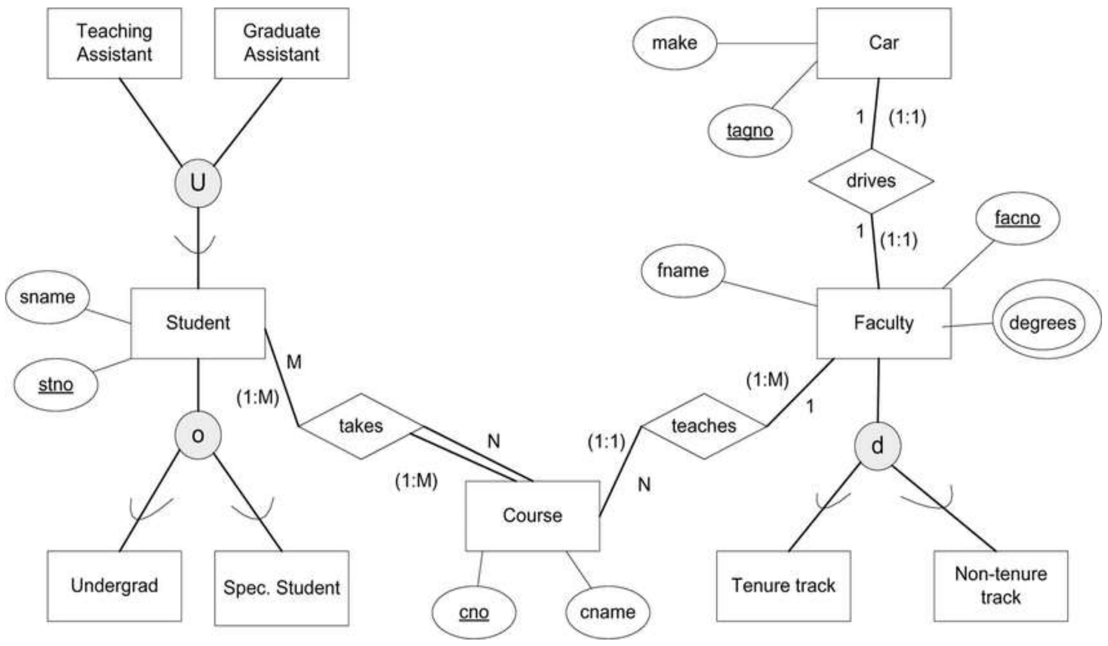
Consider an ONLINE_AUCTION database system in which members (buyers and sellers) participate in the sale of items. The data requirements for this system are summarized as follows:

- The online site has members, each of whom is identified by a unique member number and is described by an e-mail address, name, password, home address, and phone number.
- A member may be a buyer or a seller. A buyer has a shipping address recorded in the database. A seller has a bank account number and routing number recorded in the database.
- Items are placed by a seller for sale and are identified by a unique item number assigned by the system. Items are also described by an item title, a description, starting bid price, bidding increment, the start date of the auction, and the end date of the auction. For each item its category like hardware/software need to be stored.
- Buyers make bids for items they are interested in. Bid price and time of bid is recorded. The bidder at the end of the auction with the highest bid price is declared the winner and a transaction between buyer and seller may then proceed.
- The buyer and seller may record feedback regarding their completed transactions. Feedback contains a rating of the other party participating in the transaction (1–10) and a comment.

Solution



3. Convert the following EER diagram into appropriate relation with necessary constraints



Solution

```
CREATE TABLE Student (
   stno INT PRIMARY KEY,
   sname VARCHAR(100) NOT NULL
);

CREATE TABLE Undergrad (
   stno INT PRIMARY KEY,
   FOREIGN KEY (stno) REFERENCES Student(stno)
);
CREATE TABLE Specialized_Student (
   stno INT PRIMARY KEY,
   specialization VARCHAR(100),
   FOREIGN KEY (stno) REFERENCES Student(stno)
);
CREATE TABLE Teaching_Assistant (
   stno INT PRIMARY KEY,
   office_hours VARCHAR(50),
   FOREIGN KEY (stno) REFERENCES Student(stno)
);
CREATE TABLE Graduate_Assistant (
   stno INT PRIMARY KEY,
   research_topic VARCHAR(100),
   FOREIGN KEY (stno) REFERENCES Student(stno)
);
CREATE TABLE Course (
   cno INT PRIMARY KEY,
   cname VARCHAR(100) NOT NULL
);

CREATE TABLE Faculty (
   facno INT PRIMARY KEY,
   fname VARCHAR(100) NOT NULL,
   degree VARCHAR(100)
);
CREATE TABLE Tenure_Track_Faculty (
   facno INT PRIMARY KEY,
   tenure_start_date DATE,
   FOREIGN KEY (facno) REFERENCES Faculty(facno)
);
CREATE TABLE Non_Tenure_Track_Faculty (
   facno INT PRIMARY KEY,
   contract_end_date DATE,
   FOREIGN KEY (facno) REFERENCES Faculty(facno)
);
```

```
CREATE TABLE Car (
   tagno VARCHAR(20) PRIMARY KEY,
   make VARCHAR(50)
);
```

Relationship Tables

```
CREATE TABLE Takes (
   stno INT,
   cno INT,
   PRIMARY KEY (stno, cno),
   FOREIGN KEY (stno) REFERENCES Student(stno),
   FOREIGN KEY (cno) REFERENCES Course(cno)
);
CREATE TABLE Teaches (
   cno INT,
   facno INT,
   PRIMARY KEY (cno),
   FOREIGN KEY (cno) REFERENCES Course(cno),
   FOREIGN KEY (facno) REFERENCES Faculty(facno)
);
```

Drives Table (One-to-One Relationship between Faculty and Car)

```
CREATE TABLE Drives (
   facno INT PRIMARY KEY,
   tagno VARCHAR(20) UNIQUE,
   FOREIGN KEY (facno) REFERENCES Faculty(facno),
   FOREIGN KEY (tagno) REFERENCES Car(tagno)
);
```

4. Consider the following relations.

Customer (CNo, CName, Address, Phone)

Loan (LNum, Cus_No, Amount, type, Duration, sanctioned_date)

Account (Acc_Num, Cus_No, Balance)

Write relation algebra expressions for

Solution

a. Find the customer names who live in vellore.

$\pi_{\text{name}} (\sigma_{\text{address like '\%vellore\%'}} (\text{Customer}))$

b. List the number of customers in each loan.

${}_{\text{num}}\mathcal{F}\text{num, count(*)} \ (\text{Loan})$

c. Find the customers who do not have any loan.

R1 $\leftarrow \pi_{No}$ (Customer)) - π_{No} (Loan))

R2 $\leftarrow$ **R1 * Customer**

d. List the customer names who have all loan types.

R1 $\leftarrow \pi_{\textbf{distinct}\ type}$ (Loan)

R2 $\leftarrow \pi_{\textbf{cus\_no},\ type}$ (Loan)

R3 $\leftarrow$ R2 $\div$ R1

R4 $\leftarrow \pi_{name}$ (Customer * R3)

e. Find the customers who have two loans and three accounts.

R1(cus_no,nol) $\leftarrow$ $_{cus\_no}\mathcal{F}$ cus_no, count(*) (Loan)

R1(cus_no,noa) $\leftarrow$ $_{cus\_no}\mathcal{F}$ cus_no, count(*) (accounts)

R3 $\leftarrow$ ($\sigma_{nol=2 \text{ and } noa=3}$ (Customer*R1*R2))

5. Based on the above relations, you are required to give the suitable SQL statements for the following questions.

a. Create the loan and customer table with appropriate keys.

b. Print the customer number who has maximum loan (i.e. highest loan sanctioned).

c. Print the loan number and sanctioned date of given customer number.

d. Add a column type in account. Write an interactive query to put values to this column.

Solution

a. Create the loan and customer table with appropriate keys.

i) Create table customer(

No number(10),

Name varchar(20),

Address varchar(20),

Phone number(10),

Constraint customer_pk primary key(no));

ii) Create table loan

(num number(10),

Cus_no number(10),

Amount number(10,2),

Type varchar(20),

Duration varchar(10),

Sactioned_date date,

Constraint loan_pk primary key(num),

Constraint loan_fk foreign key(cus_no) references customer(no));

b. Print the customer number who has maximum loan.

   Select customer , max (amount) from customer;

c. Print the loan number and sanctioned date of given customer number. Date in the format 2nd February 2023.

   Select num, to_char(sactioned_date,'dd nd Month YYYY') from loan where cus_no=&num;

d. Add a column type in account. Write a interactive query to put values to this column.

   Alter table account add type varchar(20);

   Update account set type ='&type' where num=&num;