ITMD 523	***Advanced Database Management***	**Homework 4**
Student Name	**Harshal Sawant**	Due Date 10/28/23
Instructor	**Luke Papademas**	Section 05

Part	**1**	**2**	**3**	**4**	**Total**
maximum	**25** points	**25** points	**25** points	**25** points	**100** points
Your Score					

Topic: Advanced Data Modeling

Reading Assignment: Thoroughly read Chapter 5 in the course textbook.

Part 1 Glossary Terms

Review, in detail, each of these glossary terms from the realm of computer database systems and computer topics, in general. If applicable, use examples to support your definitions. Consult your notes and / or course textbook(s) as references.

(a) EER Diagram

EER Diagram:

An enhanced entity-relationship diagram, or EER Diagram (EERD), is a graphic depiction of a data model that applies the ideas of the Enhanced Entity-Relationship Model (EERM) or Extended Entity-Relationship Model. Entity supertypes, entity subtypes, and entity clustering are some of the new semantic components that this paradigm adds to the conventional Entity-Relationship (ER) model. EER diagrams are useful for creating databases for systems that have complicated data needs because they can more fully depict complex data structures and their interactions.

Supertypes and Subtypes of Entities:

Entity supertypes and subtypes are fundamental ideas in the Extended Entity Relationship Model (EERM) that help structure and arrange data entities. Higher-level entities known as entity supertypes generalize shared characteristics among related subtypes. On the other hand, entity subtypes can contain unique properties in addition to inheriting traits from supertypes.

Example:

"Employee" can be the entity supertype in a company's database, signifying common properties such as "Name" and "Hire Date." The entity subtypes "Manager" and "Technician" may inherit these properties as well as possess unique attributes such as "Team" and "Manager ID" for managers and "Skills" and "Technician ID" and "Manager ID" for technicians. Relationships specific to each subtype are facilitated and data duplication is reduced by this structure.

(b) Entity Cluster

Entity Cluster:

An entity cluster is a fundamental idea in data modelling and computer database systems, mostly used in the construction of entity-relationship diagrams (ERD). It performs the crucial function of streamlining sophisticated ERDs that might include a large number of entities and complex interactions. An entity cluster is essentially an abstract or virtual entity type that is used to conceptually represent numerous connected entities but does not exist in the final ERD. Improving the ERD's readability and clarity is the primary goal.

An "PROGRAM" entity cluster, for **example**, can represent the relationships between entities like "SEMESTER," "COURSE," and "CLASS" in a university database system. It's important to keep in mind, though, that utilizing entity clusters may prevent specific entity properties from being shown, which might have an impact on important attributes and inheritance rules. In order to make the diagram more simplified at the expense of attribute visibility, it is therefore frequently advised against displaying attributes within entity clusters. To guarantee an efficient and understandable data model, this trade-off must be taken into account during the database design process.

(c) Fan Trap

Fan Trap:

In the context of database systems and entity-relationship modeling (ERD), a particular kind of design problem that might arise during the definition of relationships between entities is referred to as a "fan trap." It's a type of design pitfall that might result in a database that inaccurately or inconsistently represents the connections in the actual world. The primary feature of a fan trap is when one entity has two or more one-to-many (1:M) links with other entities, which leads to an unclear or poorly defined relationship between the other entities. It might be challenging to produce insightful reports or precisely extract data from the database as a result of this uncertainty.

Example can be:

When an entity, like DIVISION, is related to several other entities (like TEAM and PLAYER) via distinct one-to-many (1:M) connections, it can lead to a fan trap in database architecture. This makes it difficult to grasp how the many entities relate to one another. In a basketball league database, for instance, it would be difficult to determine which players are on which teams if DIVISION was mistakenly connected to both TEAM and PLAYER. Redesigning the database will remove ambiguity and ensure proper data representation. Division is tied to TEAM by a 1:M relationship, and TEAM is related to PLAYER via another 1:M relationship. This will fix fan traps. In order to avoid data discrepancies and enable precise querying and reporting, fan traps should be taken into consideration during database architecture.

(d) Generalization and Specialization

Generalization:

A specific kind of relationship in a database, known as a 1:1 (one-to-one) relationship, is called a generalization. It denotes a direct relationship between one entity and another, and vice versa. As an illustration, a department is headed by a professor, and a department has a professor as its chair. Foreign keys are used to implement this 1:1 relationship and show the relationships between the entities. It is a technique for guaranteeing data integrity and consistency in databases, but it should be applied carefully as it might result in intricate designs. Under some circumstances, generalization hierarchies—including 1:1 relationships—can be helpful for enhancing database architectures.

Specialization:

ITMD 523	*Advanced Database Management*	Homework 4
Student Name	**Harshal Sawant**	Due Date 10/28/23
Instructor	**Luke Papademas**	Section 05

In computer database systems, specialization is arranging entities into a hierarchy in which subtypes, or lower-level things, are connected to higher-level entities (supertypes). In a business database, for example, you may have subtypes such as MANAGER, TECHNICIAN, and ACCOUNTANT under the supertype EMPLOYEE. Subtypes have distinct qualities, while entity supertypes show the shared traits of subtypes. Because of this arrangement, characteristics can be inherited by subtypes from supertypes. A subtype discriminator aids in determining an instance's subtype. Disjoint or overlapping constraints, which specify whether an instance can belong to one or more subtypes, are another aspect of specialization. Similarly, whether a subtype is required for each supertype instance depends on whether constraints are complete or partial. By effectively handling different qualities and organizing data relationships, specialization hierarchies enhance database models.

(e) Subtype Discriminator

Subtype Discriminator:
In the framework of entity-relationship modeling, which is frequently utilized in database architecture, a subtype discriminator is a crucial idea. It is essential for figuring out which subtype (a specialized or sub-category) a particular instance of a supertype object falls into.
In entity-relationship modeling, a "subtype discriminator" is a crucial idea that helps classify instances of a supertype entity into different subtypes based on the value of a given property. This approach maintains the distinct relationships and properties of each category while allowing the depiction of many data categories within a single entity. According to this method, a subtype is a more specialized version of the "supertype" object. Under the "EMPLOYEE" supertype, for instance, the subtypes "MANAGER," "TECHNICIAN," and "ACCOUNTANT" exist. An employee's role is determined by the subtype discriminator, the "EMP_TYPE" property. This fundamental idea of database design makes sure that the structure of data complies with business needs and real-world circumstances.

Part 2 Completion: True / False Exercises

For each of these exercises, enter either True or False.

(1) **TRUE** The relationships depicted within the specialization hierarchy are sometimes described in terms of "is-a" relationships.

(2) **FALSE** Within a specialization hierarchy, a supertype can exist only within the context of a subtype.

(3) **FALSE** A subtype contains attributes that are common to all of its supertypes.

(4) **TRUE** The entity supertype contains common characteristics, and the entity subtypes each contain their own unique characteristics.

(5) **FALSE** In specialization hierarchies with multiple levels of supertype and subtypes, a lower-level subtype can inherit only a few of the attributes and relationships from its upper-level supertypes.

Part 3 Multiple Choice Exercises

Select the correct answer.

(1) What type of subtypes are subtypes that contain nonunique subsets of the supertype entity set?

a. Disjoint **b. Overlapping**
c. Completeness d. Discriminator

(2) What do you add to the supertype table for a disjoint condition?

a. disjoint discriminator **b. subtype discriminator**
c. supertype discriminator d. joint discriminator

(3) What is the top-down process of identifying lower-level, more specific entity subtypes from a higher-level entity supertype?

a. specialization
b. inheritance
c. relationship
d. generalization

(4) If we are creating subtypes for a car dealership. Why would an accountant be a subtype of an employee?

a. because it is identifiable with employees and has unique attributes
b. because it only is identifiable with employees
c. because it is not identifiable with employees but has unique attributes
d. because it only has unique attributes

(5) Within a specialization hierarchy, every subtype can have _____ supertype(s) to which it is directly related.

a. zero **b. only one**

c. one or many d. many

Part 4 Essays Exercises

Write a brief but complete answer to each of these questions / exercises.

(1) Explain when you would use Composite Primary Keys. Define surrogate key and when to use it. Based on what you know about primary keys, what can be said about the "no change over time"?

Composite Primary Keys:

When one column in a database structure is unable to uniquely identify every record in a table, Composite Primary Keys are utilized. Instead, a unique identification is created by combining numerous columns. When distinctness cannot be ensured by individual columns alone, this is required.

Surrogate Key:

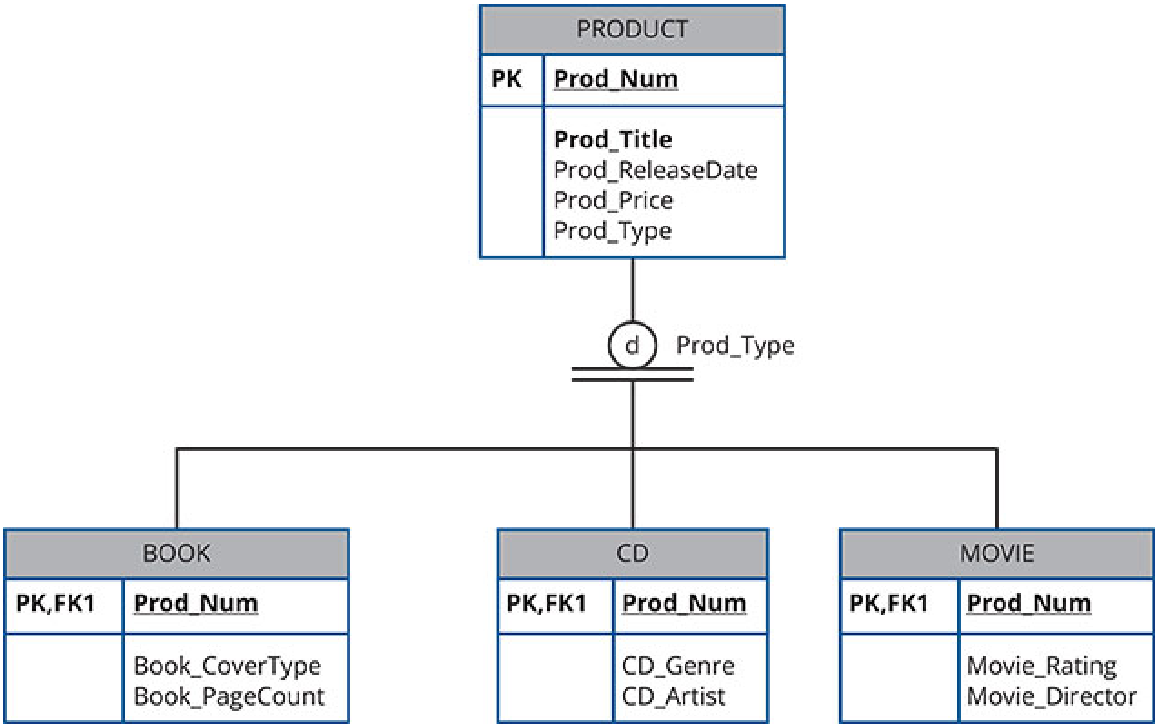
In order to guarantee that every entry in a database has a unique identifier, a fake primary key—typically an auto-incremented integer—is inserted as a surrogate key. When there isn't a good natural key or when the natural key might vary over time, it's frequently employed. Even in the event that the data they refer to changes, surrogate keys remain constant.

No Change Overtime:

With respect to the notion of "no change over time," this means that a primary key—natural or substitute—should be true throughout the duration of a record. Ensuring the consistency and integrity of data is crucial. A change in the primary key would cause data anomalies, which would compromise referential integrity and make it challenging to trace past data or build connections with other types of data. As a result, stability and uniqueness should be guaranteed by carefully selecting the main keys.

(2) Examine the following Extended Entity Relationship Model (EERM) .

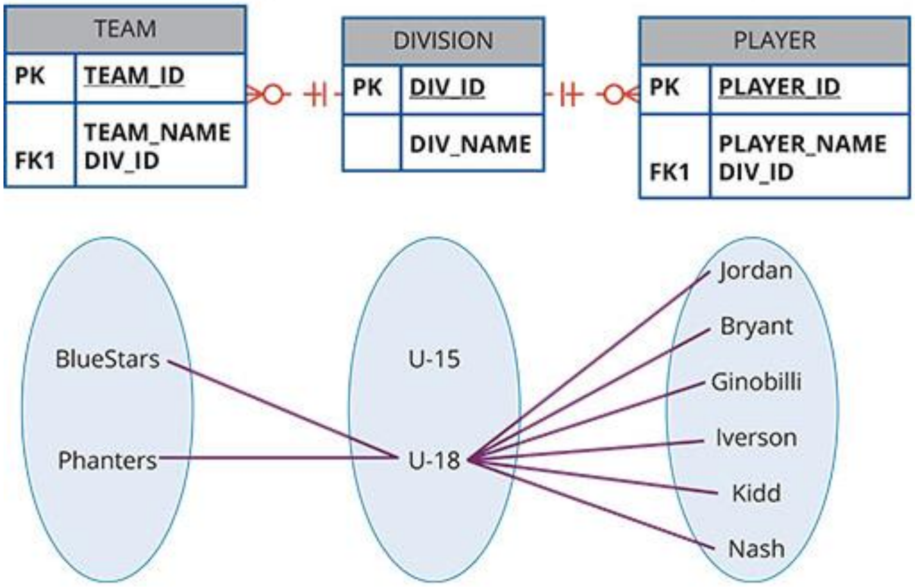
ITMD 523	***Advanced Database Management***	**Homework 4**
Student Name	**Harshal Sawant**	Due Date 10/28/23
Instructor	**Luke Papademas**	Section 05



Using glossary terms, concepts and procedures that are outlined in the course textbook, explain, in detail, what this EERM attempts to describe.

Three different product kinds are described in the EERM above: books, CDs, and movies. "PRODUCT" is the common entity throughout all product classes. As a discriminator—a primary key Prod_Num—the PRODUCT entity may be used to distinguish between three different kinds of goods. PRODUCT entity is inherited in all the other entities with disjoint subtype. Book_CoverType, Book_PageCount, and Prod_Num, a primary and foreign key, are all part of the BOOK object. On the other hand, CD_Artist, CD_Genre, and Prod_Num are the foreign and main keys in the CD object. Additionally, Prod_Num, Movie_Rating, and Movie_Director are both main and foreign keys in the MOVIE entity. The BOOK, CD, and MOVIE entities are subtypes of the PRODUCT entity, and they have a one-to-one connection with the PRODUCT entity. A superclass-subclass link between the PRODUCT entity and its subtype entities is also shown in this EERM.

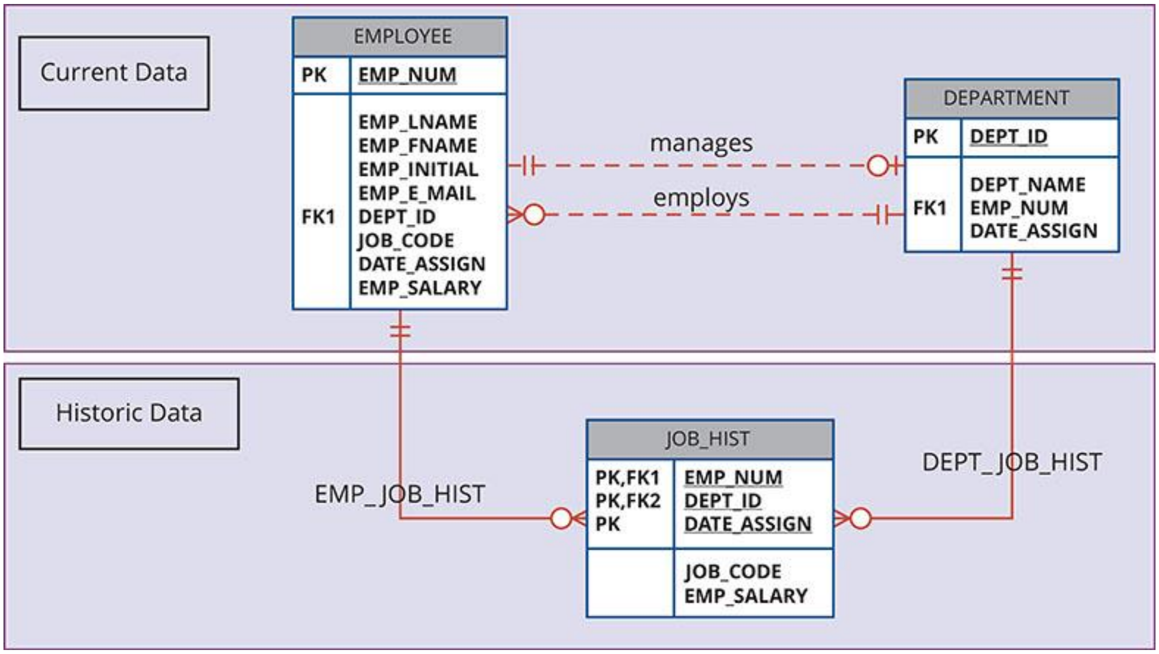
(3) Examine the entity relationship and explain your steps to eliminate the apparent fan trap.



A fan trap separates the PLAYER and TEAM entities in the ER diagram above. Two one-to-many connections exist between these entities, which is the cause of this. Many players can be on one team, and a player can only be on one team. We must divide the division entity into two entities in order to get rid of the fan trap. The ER diagram can be normalized to achieve this. A one-to-many link is established between each new entity and the PLAYER entity after the Division has been divided into U15_DIVISION and U18_DIVISION. We have a new, normalized ER diagram and have removed the fan trap after deleting the two one-to-many links between the TEAM and PLAYER entities.

(4) Using glossary terms, concepts and procedures that are outlined in the course textbook, explain, in detail, what this data model attempts to convey.

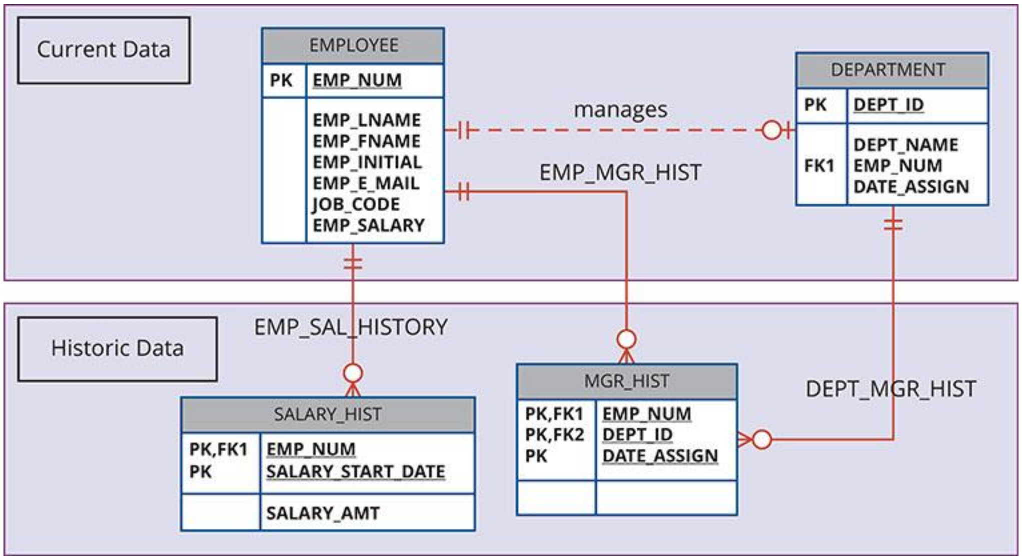
ITMD 523	*Advanced Database Management*	Homework 4
Student Name	**Harshal Sawant**	Due Date 10/28/23
Instructor	**Luke Papademas**	Section 05



An EERM graphic showing historical and current employee statistics is shown above. Three entities—EMPLOYEE, JOB_HIST, and DEPARTMENT—are shown in the diagram. These entities are tied to one another; an employee may be assigned to several jobs, but they may only belong to one department. There are two types of keys: one for the employer, EMP_NUM, and a foreign key called DEPT_ID; for the department, DEPT_ID is the main key and EMP_NUM is the foreign key. The link between the employee and the department is one-to-many, whereas the relationship between the employee and Job History is many-to-many. The JOB_HIST entity belongs to the EMPLOYEE entity subtype. As a composite entity with a composite primary key, the JOB_HIST object is constructed from the EMPLOYEE and DEPARTMENT entities. The EMPLOYEE and JOB_HIST entities establish a superclass-subclass connection in which the latter will inherit all of the former's qualities in addition to some of its own.

(5) Using glossary terms, concepts and procedures that are outlined in the course textbook, explain, in detail, what this data model attempts to convey.

ITMD 523	*Advanced Database Management*	Homework 4
Student Name	**Harshal Sawant**	Due Date 10/28/23
Instructor	**Luke Papademas**	Section 05



Above is an ERD for an employee database. The ERD has four entities: MGR_HIST, SALARY_HIST, DEPARTMENT, and EMPLOYEE. These entities each have a primary key, a few foreign keys, and a unique collection of properties. Data about the employees, such as their job code, email address, first and last names, initials, and compensation, are recorded using the primary key EMP_NUM of the EMPLOYEE object. The department name, assigned data, and employee number (a foreign key) are all contained in the DEPARTMENT entity. DEPT_ID is the main key for it. The SALARY_HIST object contains the pay history for each employee, including their employee number, start date, salary, and manager. A single employee can oversee one or zero departments in a one-to-zero or one-to-one relationship with the department. Employees can have many salary histories or none at all in the one-to-many or zero connection between EMPLOYEE and SALARY_HIST. A single employee is linked to several management history records in a one-to-many or zero connection between EMPLOYEE and MGR_HIST. Additionally, there is a one-to-many or zero link between DEPARTMENT and MGR_HIST, meaning that a department may contain several management history records.