Journal A – Database Systems Omar Salih Stu#: 12285240

**1a. Determinant (note this is the most important concept in the course). Provide two illustrations of the concept with real word data attributes (not X, Y etc.). In examples list the determinant and the dependents as well.**

**Definition:**

A determinant in a database table is any attribute that you can use to determine the values assigned to other attribute(s) in the same row.

Note that a determinant can be a Field (FD) or of a table. A primary key or any candidate key is also a determinant while the opposite is not true. A CK can only be of a table. Every CK is a determinant *of “its table”*. A determinant can uniquely determine one or more attributes in the row. A candidate key can uniquely determine the entire row.

**Example 1:**

**TABLE**

Employee (employee\_id, first\_name, last\_name, date\_of\_birth)

In this case, the field employee\_id determines the remaining three fields. The name fields do not determine the employee\_id because the firm may have more than one employee with the same first and/or last name. Similarly, the DOB field does not determine the employee\_id or the name fields because more than one employee may share the same birthday.

**Example 2:**

**TABLE**

US\_Address (AddressID int, Streetline, City, State, ZIP, StateName, StateTax)

State is a determinant for StateName and StateTax, but it is not a candidate key for the row. Proper normalization, would therefore move StateName and StateTax out of the US\_Address table and into a States table.

**1b. Difference between Super key and Primary Key. Provide an example of each.**

**Primary Key**

Primary Key is a minimal field or candidate key, or a combination of fields within a record, that is chosen, because it maintains uniqueness in a table at the record (row) level, to identify entities with in an entity set OR is used to uniquely identify each record (row) OR a minimal field within the record which is sufficient to identify the row is a primary key.  Primary key is subset of super key but vice versa is not true.

Only one Candidate Key can be a Primary Key. For a Candidate Key to qualify as a Primary Key, it should be unique and a non-null. So, basically a primary key is just one of the candidate keys, which is a just a minimal super key.

A primary key, also known as primary keyword, is a key in a relational database that is unique for each record. It is a unique identifier, such as a driver license number, telephone number (including area code), or vehicle identification number (VIN). A relational database must always have one and only one primary key.

**Super Key**

The concept of a Super Key maps the concept of a *unique non-clustered index* with additional columns for improved query covering. A Super key is used for identifying a row. It may contain one or more Primary Key combination(s), known as a Super Key. A *Super Key* is simply a non-minimal *Candidate Key*, that is to say one with additional columns not strictly required to ensure uniqueness of the row.

Super key stands for superset of a key. A Super Key is a set of one or more attributes that are taken collectively and can identify all other attributes uniquely.

For Example, we have a table

Book (BookId, BookName, Author)

So in this table we can have these keys

(BookId)

(BookId,BookName)

(BookId, Author, BookName)

(BookId, Author)

(BookName, Author)

As our Super key. Each super key is able to uniquely identify each record.

So, a super key is a set of one or more attribute fields, that can include one or more primary keys, to uniquely identify the records (tuples) values in a table.

**1c. Composite key. Provide an example.**

When a primary key is created from a combination of 2 or more columns, the primary key is called a composite key. Each column may not be unique by itself within the database table but when combined with the other column(s) in the composite key, the combination is unique.  
  
To illustrate the concept of the composite key and the foreign key, consider the sample table design below:  
  
**Customer Table**

|  |  |
| --- | --- |
| **Column** |  |
| lastname | primary key |
| firstname | primary key |
| dateofbirth |  |

The lastname column and the firstname column together form a composite key. Let's assume that the table above contains the following data:

|  |  |  |
| --- | --- | --- |
| **lastname** | **firstname** | **dateofbirth** |
| henry | John | 03/05/1960 |
| henry | Adam | 06/08/1974 |
| kidman | Adam | 04/01/1955 |
| bailey | Harry | 05/05/1980 |
| morgan | Alex | 09/09/1975 |

Notice that in the lastname column, there are 2 records with the value 'henry' and in the firstname column, there are 2 records with the value 'adam'. However, there are no records in the database table with a duplicate combination of both the lastname and the firstname.  
  
BTW, the above table design is very problematic, sooner or later you will have customers changing their names or customers with exactly the same name...

**1d. Referential Integrity.**

Referential integrity means the foreign key in any referencing table must always refer to a valid row in the referenced table. Referential integrity ensures that the relationship between two tables remains synchronized during updates and deletes.

For example, assume that your application has both a Titles table and a Publishers table as shown in the following table.

|  |  |
| --- | --- |
| **Titles table** | **Publishers table** |
| ti\_isbn (key) | pu\_id (key) |
| ti\_title | pu\_name |
| ti\_yearpublished | pu\_address |
| pu\_id  (foreign key) | pu\_phone |

Referential integrity requires that these two tables must be synchronized. That is, each publisher identification (pu\_id) in the Titles table must also exist in the Publishers table.

Your application cannot just delete the pu\_id row from the Publishers table because that would leave the pu\_id in the Titles table without a reference. It would be permissible, however, to delete the pu\_id row from the Publishers table and also delete every row in the Titles table that has the same pu\_id. Such an action would maintain referential integrity for these two tables.

In a similar manner, your application cannot just add a row to the Titles table without a valid pu\_id already in the Publishers table. To do so would insert "bad" data in the pu\_id field. So, your application must ensure a valid pu\_id key in the Publishers table before inserting the pu\_id in the related Titles row.

The actual implementation of referential integrity depends entirely on the data storage engine you choose and your application's design requirements. Historically, applications using mainframe VSAM files used application code to handle referential integrity. Today, even if your application uses SQL Server, that does not mean you must use triggers, foreign keys, constraints, and cascading deletes to maintain referential integrity. You might again choose to handle referential issues with application-based code.

**2. Difference between repeating data and redundant data. Please provide a real life example.**

**Redundant**

Data **redundancy** is a condition created within a **database** or data storage technology in which the same piece of data is held in two separate places. This can mean two different fields within a single **database**, or two different spots in multiple software environments or platforms.

Data should not be redundant, which means that the duplication of data should be kept to a minimum for several reasons. For example, it is unnecessary to store an employee's home address in more than one table. With duplicate data, unnecessary space is used. Confusion is always a threat when, for instance, an address for an employee in one table does not match the address of the same employee in another table. Which table is correct? Do you have documentation to verify the employee's current address? As if data management were not difficult enough, redundancy of data could prove to be a disaster.

Redundant data is the same data being stored in different tables.

**Repeating Data**

The Tables design should be as simple as possible by restricting each position to single entry and by preventing multiple entries, also called repeating data, in an individual location in the table:

ORDERS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Order# | OrderDate | CustomerNum | PartNum | NumbOrdered | QuotedPrice |
| 23564 | 10/20/2020 | 250 | ANZ342 | 244 | 500.00 |
| 45224 | 10/20/2020 | 246 | RR0494  RR0445 | 2  1 | 240.00  157.00 |
| 7553 | 10/21/2020 | 503 | ANZ342 | 450 | 639.00 |
| 2458 | 10/21/2020 | 455 | CD9453 | 4256 | 5,042.00 |
| 4204 | 10/22/2020 | 45 | DGI346 | 242 | 156.00 |

ORDER (Without Repeating Data)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Order# | OrderDate | CustomerNum | PartNum | NumbOrdered | QuotedPrice |
| 23564 | 10/20/2020 | 250 | ANZ342 | 244 | 500.00 |
| 35322 | 10/20/2020 | 246 | RR0494 | 2 | 240.00 |
| 45224 | 10/20/2020 | 246 | RR0445 | 1 | 157.00 |
| 7553 | 10/21/2020 | 503 | ANZ342 | 450 | 639.00 |
| 2458 | 10/21/2020 | 455 | CD9453 | 4256 | 5,042.00 |
| 4204 | 10/22/2020 | 45 | DGI346 | 242 | 156.00 |

Repeating data is data that is stored in the same field. A combination of data stored together for the same customer is an example.

**3. Please don’t use X, Y, Z or A, B, C or such abstract variables. Only real world data examples are permissible.**

**Explain subtype and provide an example.**

A subtype is an entity of a supertype entity, which a supertype may be related to one or more subtype entity. When dealing with specialization hierarchy, the subtypes only have one supertype relationship. The subtypes inherit the supertypes attributes. To determine which supertype the subtype occurs in, we have the subtype discriminator. For example, the subtype discriminator determines if it has a partial or total completeness, also is it specialization or generalization. Basically, a subgrouping of the entities in an entity type that is meaningful to the organization and that shares common attributes or relationships distinct from other subgroupings.

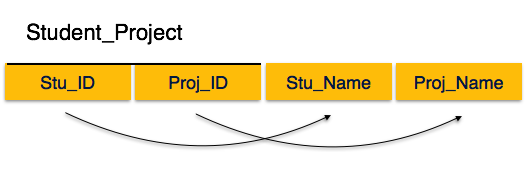
An example would be, an employee can be of subtypes of mechanic, developer, actor. Now the attributes of the superkey employee will be absorbed by all the subkeys, which have unique attributes. The inherited relationship between them can be either disjoint, overlapping, or has. The it can be a partial or complete constraint. The relationship is defined by what the subtype discriminator is.

**2. Explain these terms with an example for each:**

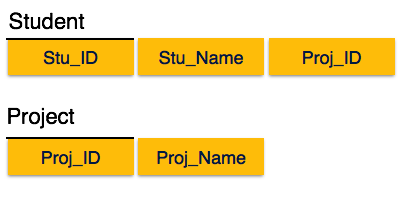
**a. Partial Dependency**

Partial Dependency is a form of Functional dependency that holds on a set of attributes. It is about the complete dependency of a right hand side attribute on one of the left hand side attributes. In a functional dependency XY → Z, if Z (RHS attribute) can be uniquely identified by one of the LHS attributes, then the functional dependency is partial dependency.

If we follow second normal form, then every non-prime attribute should be fully functionally dependent on prime key attribute.



We see here in Student\_Project relation that the prime key attributes are Stu\_ID and Proj\_ID. According to the rule, non-key attributes, therefore Stu\_Name and Proj\_Name must be dependent upon both and not on any of the prime key attribute individually. But we find that Stu\_Name can be identified by Stu\_ID and Proj\_Name can be identified by Proj\_ID independently. This is called partial dependency, which is not allowed in Second Normal Form.



Broke the relation in two as depicted in the above picture. So there exists no partial dependency.

**b. Repeating Data**

* Group of multiple entries of same type can exist for any single key attribute occurrence

ORDERS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Order# | OrderDate | CustomerNum | PartNum | NumbOrdered | QuotedPrice |
| 23564 | 10/20/2020 | 250 | ANZ342 | 244 | 500.00 |
| 45224 | 10/20/2020 | 246 | RR0494  RR0445 | 2  1 | 240.00  157.00 |
| 7553 | 10/21/2020 | 503 | ANZ342 | 450 | 639.00 |
| 2458 | 10/21/2020 | 455 | CD9453 | 4256 | 5,042.00 |
| 4204 | 10/22/2020 | 45 | DGI346 | 242 | 156.00 |

Order number 45224 PartNum, NumOrderd, QuotedPrice is an example of repeating data.

**c. Transitive Dependency**

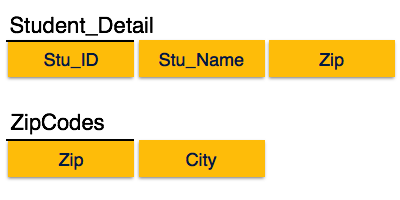
For a relation to be in Third Normal Form, it must be in Second Normal form and the following must satisfy

* No non-prime attribute is transitively dependent on prime key attribute.
* For any non-trivial functional dependency, X → A, then either X is a superkey or A is prime attribute.



We find that in the above Student\_detail relation, Stu\_ID is the key and only prime key attribute. We find that City can be identified by Stu\_ID as well as Zip itself. Neither Zip is a superkey nor is City a prime attribute. Additionally, Stu\_ID → Zip → City, so there exists transitive dependency.

To bring this relation into third normal form, we break the relation into two relations as follows



**3. On which side would a foreign key go in a parent child (1:M) relationship?**

Right side. Child side.

**4. What is an associative entity? Please provide an example of how M:N relationship can be handled in the design process through a bridge/associative entity? Provide an illustration different from the movie and star relationship discussed in the class. Make an ERD for your example.**

An [associative entity](http://en.wikipedia.org/wiki/Associative_entity) is the table that associates two other tables in a many to many relationships.

An associative relationship attribute is an attribute of the associative entity that exists because of the many to many relationships.

Here's an example. Let's suppose we have the following tables

User

User ID

User Login Name

User Name

User Password

Permission

Permission ID

Permission Name

Permission Description

We have a many to many relationships between User and Permission. A user can have more than one permission, and a permission can be shared between many users.

So, we create an associative entity.

User Permission

User ID

Permission ID

Permission Granted Time Stamp

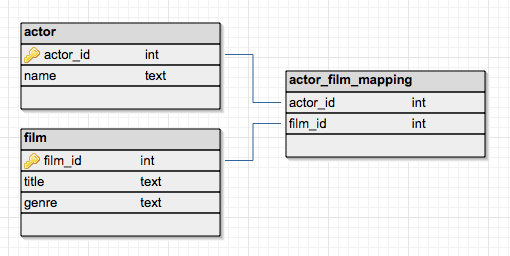
The permission granted time stamp is an associative relationship attribute. It would not fit in the User table nor the Permission table. It's an attribute of the association.

-----

A relational database requires the implementation of a base relation, base table, to resolve [many-to-many relationships](https://en.wikipedia.org/wiki/Many-to-many_(data_model)). This kind of base relation is called an associative table.

[https://upload.wikimedia.org/wikipedia/en/1/14/Associate_Entity.png](https://en.wikipedia.org/wiki/File:Associate_Entity.png)

associative entities are implemented in a database structure using associative tables, which are tables that can contain references to columns from the same or different database tables within the same database.

[](https://en.wikipedia.org/wiki/File:Mapping_table_concept.png)

An associative table maps two or more tables together by referencing the primary keys of each data table. In effect, it contains a number of foreign keys, each in a many-to-one relationship from the associative table to the individual data tables. The PK of the associative table is typically composed of the FK columns themselves.

**5. What is a weak entity? Provide an example of how repeating data can be handled by creating a weak entity. We have learned how to handle the case of major, a repeating attribute in Student table, therefore, please provide a different example. Make an ERD for your example.**

A weak entity is an entity that must defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone.

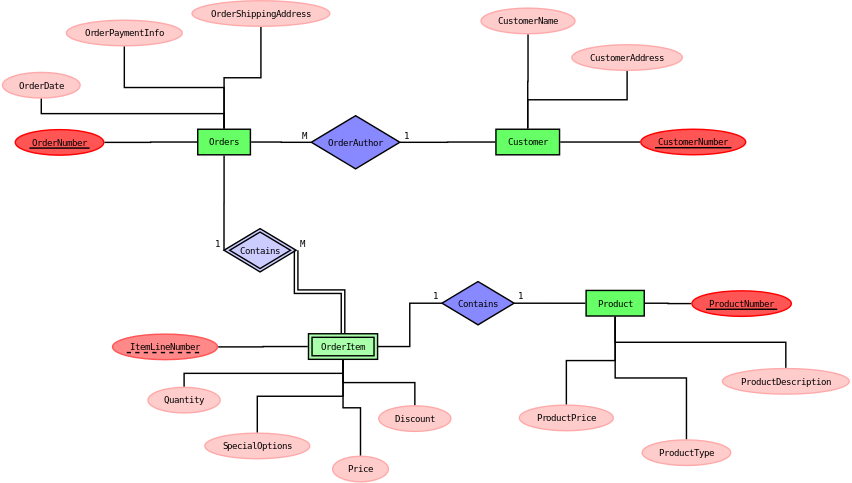
Weak entity is an entity that cannot be uniquely identified by its attributes alone; therefore, it must use a [foreign key](https://en.wikipedia.org/wiki/Foreign_key) in conjunction with its attributes to create a [primary key](https://en.wikipedia.org/wiki/Primary_key). The foreign key is typically a primary key of an entity it is related to.

In [entity relationship diagrams, ER diagrams](https://en.wikipedia.org/wiki/Entity_relationship_diagram) a weak entity set is indicated by a bold (or double-lined) rectangle (the entity) connected by a bold (or double-lined) type arrow to a bold (or double-lined) diamond (the relationship). This type of relationship is called an identifying relationship and in [IDEF1X](https://en.wikipedia.org/wiki/IDEF1X) notation it is represented by an oval entity rather than a square entity for base tables. An identifying relationship is one where the primary key is populated to the child weak entity as a primary key in that entity.

In general, (though not necessarily) a weak entity does not have any items in its primary key other than its inherited primary key and a sequence number. There are two types of weak entities: [associative entities](https://en.wikipedia.org/wiki/Associative_Entities) and [subtype entities](https://en.wikipedia.org/w/index.php?title=Subtype_entities&action=edit&redlink=1). The latter represents a crucial type of [normalization](https://en.wikipedia.org/wiki/Database_normalization), where the [super-type entity](https://en.wikipedia.org/w/index.php?title=Super-type_entity&action=edit&redlink=1) inherits its attributes to entities based on the value of the [discriminator](https://en.wikipedia.org/wiki/Discriminator).

EXAMPLE:

Consider a database that records customer orders, where an order is for one or more of the items that the enterprise sells. The database would contain a table identifying customers by a customer number ([primary key](https://en.wikipedia.org/wiki/Primary_key)); another identifying the products that can be sold by a product number ([primary key](https://en.wikipedia.org/wiki/Primary_key)); and it would contain a pair of tables describing orders.

[](https://en.wikipedia.org/wiki/File:Weak_entity_ER-example.svg)

References Uses:

<https://en.wikipedia.org>, ecomputernotes.com, [www.datanamic.com](http://www.datanamic.com), <https://opentextbc.ca>, stackoverflow.com, databases.about.com, [www.pcmag.com](http://www.pcmag.com), <https://exploredatabase.blogspot.com/>, [Reference.com](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0ahUKEwi5yb7W4sHPAhUl2IMKHSeEB48QFggnMAI&url=https%3A%2F%2Fwww.reference.com%2Ftechnology%2Fpartial-dependency-91beb2fa8c8354c5&usg=AFQjCNHZH9O5LLSvAIblDp_S7uH-7X8WDw&sig2=uNodGPAiSF4zWJ8WmZyHvw), [www.sciencedirect.com](http://www.sciencedirect.com), [www.prenhall.com](http://www.prenhall.com), <https://learndatamodeling.com>, <https://msdn.microsoft.com>, sqlmag.com, support.esri.com, geverest.umn.edu/, dba.stackexchange.com, [www.webopedia.com](http://www.webopedia.com)