Module_3: Mapping ER to Schema, Normalization

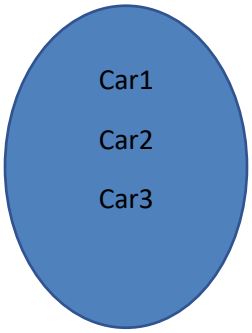
3.1 Introduction and Background

Mapping an Entity Relationship (ER) model gives a good overview of the design of a system with the goal of making the system easier to understand at a technical level. The ER diagrams can be mapped to a relation schema, which means we can clearly display the relationship between its members. The database **schema** is a structure that describes in a formal language the association of data as a blueprint of how the database can be constructed.

To understand this process, we will review below what an ER model is. An ER model is used to illustrate the logical interpretation of the system and consists of three components: Entity, Entity Type and Entity Set.

Entity can refer to a person, a concept, an object, a virtual file, or it can represent an idea that can be quantified, such as a company, a job, or a document. An entity consists of an Entity Type and everything that consists of an entity type is called an Entity Set. The example below will clarify the power of design a schema showcasing an entity type, “Car,” and an entity set consisting of all cars.

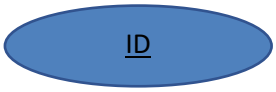
Entity Type



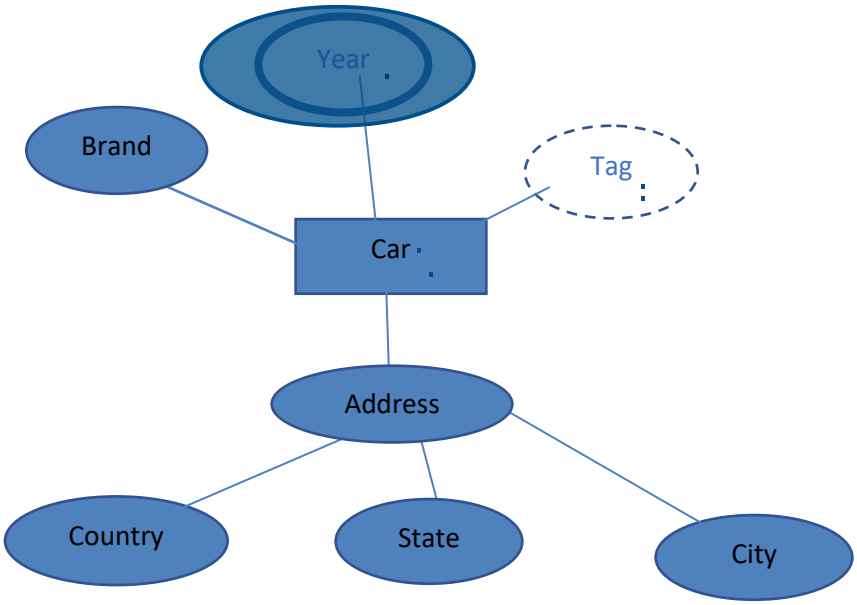
Entity Set

The next important step in understanding the mapping is an Attribute. Attributes are properties which define the entity type. For example, a person may have the attributes name, date of birth, and address. The attribute in an ER diagram is represented by an oval.

The attribute that uniquely identifies each entity is called a Key Attribute and is represented by an oval with underlining. A key attribute can be “ID,” which identifies each entity by a unique number.

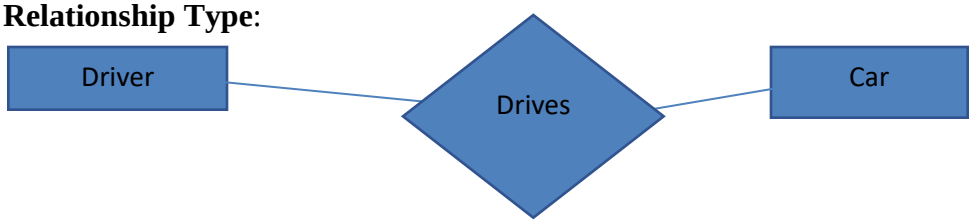


For complex designs, there are multiple different types of attributes: Composite Attributes, consisting of many individual attributes, Multivalued Attributes, consisting more than one value, and Derived Attributes, which are derived from other attributes as the name suggests.

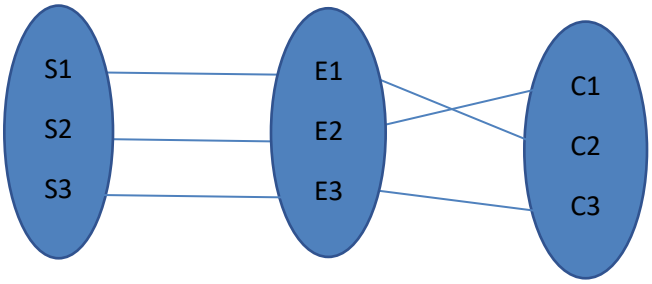


Between entities, there are relationships. A relationship has a type and can be part of a sets. The latter consists of one, two or many relationship types: unary, binary or N-ary.

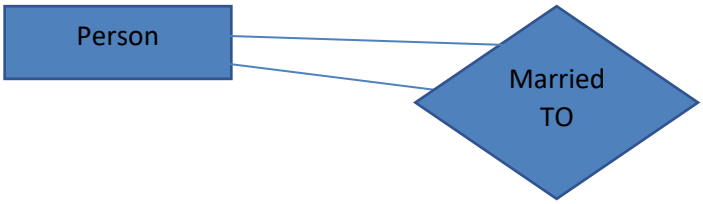
Relationship Type:



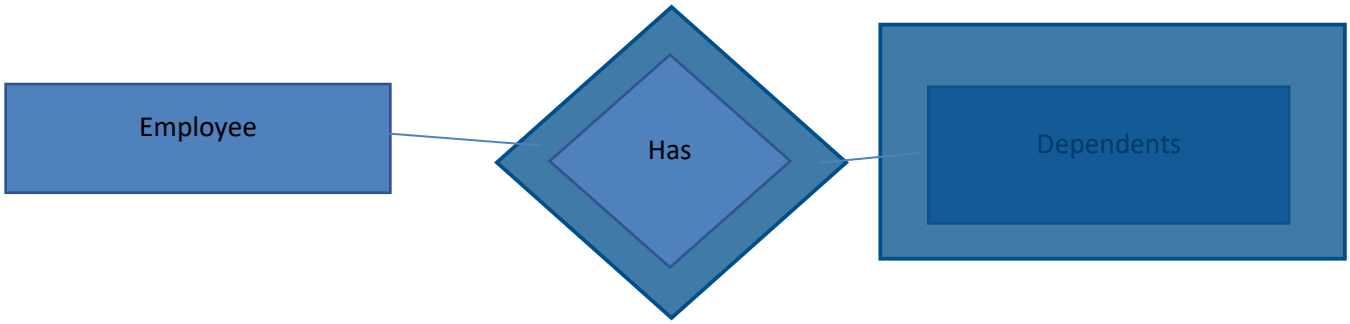
Relationship Set:



Unary, Binary Relationship:



An ER diagram can also have a Weak Entity. A weak entity is an entity type for which a key attribute can't be defined. A company can store information of dependents (Parents, Children, Spouse) of an Employee but the dependents don't exist without the Employee; we call this entity a weak entity.



3.2 List five properties of relations

In database systems, there exist relations in a two-dimensional table of data. This typically has multiple named columns and an indeterminate quantity of rows for data to be entered. A common misconception is a database relation being mistaken for a relational database. The main point to remember is that a relation specifically refers to an individual table in a relational database. There are specific properties that define relations so that they can be easily identified by a user. These relations have many properties that identify and separate themselves from non-relational tables.

Properties of relations:

1. Each table in a database has a unique identity (name).
2. Any entry at the connection of each row and column has a single value. There can be only one value that is related with each attribute on a specific row of a table; no multivalued attributes are allowed in a relation.
3. Each row is unique; no two rows or tables in the same relation can be identical.
4. Each attribute (or column) within a table has a unique name.
5. The sequence of columns (left to right) is insignificant. The order of the columns in a relation can be changed without changing the meaning or use of the relation.
6. The sequence of rows (top to bottom) is insignificant. As with columns, the order of the rows of a relation may be changed or stored in any sequence.

A few of these properties tie into one another, namely, the first, fourth, and sixth. The columns having unique attributes and the individual rows being arbitrarily organized very simply explains the core properties of relations.

EmpID	Name	DeptName	Salary	CourseTitle	DateCompleted

Figure 1 Example of property one. Referenced material from Fig 4-2 in book.

The second property speaks to Codd's original notion from 1969 that each and every attribute in individual tuples inside of a relation must only consist of a single value and not allow any different multivalued values of the kind supported in databases.

EMPLOYEE1

EmpID	Name	DeptName	Salary
100	Margaret Simpson	Marketing	48,000
140	Allen Beeton	Accounting	52,000
110	Chris Lucero	Info Systems	43,000
190	Lorenzo Davis	Finance	55,000
150	Susan Martin	Marketing	42,000

Figure 2 Example of property two. Referenced material from Fig 4-1 in book.

The third property speaks to the uniqueness within the table itself and how each individual row can only have values of data that are unique to the row itself. This ensures that no two rows will be identical.

EmpID	Name	DeptName	Salary	CourseTitle	DateCompleted
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/2015
				Surveys	10/7/2015
140	Alan Beeton	Accounting	52,000	Tax Acc	12/8/2015
110	Chris Lucero	Info Systems	43,000	Visual Basic	1/12/2015
				C++	4/22/2015
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/16/2015
				Java	8/12/2015

Figure 3 - Example of property three. Referenced material from Fig 4-2 in book.

The fifth property specifically describes how the order does not matter in regard to the columns themselves. In figure 3, the order of columns can be arbitrary in the structure itself.

Entities, Relations, and Characteristics

- Each row contains data about an entity (table).
- Each column contains data related to attributes of the entities.
- All entries in a column are of the same kind.
- Each cell of the table holds a single value.
- The order of the columns is not important.
- The order of the rows is not important
- No two rows may be identical.

EmpolyeerNumber	FirsName	LastName	Department	EmailAddress	PhoneNum
10	Judy	Griffin	Payroll	jg@gmail.com	780-492-5689
20	Lily	Murphy	Security	lm@gmail.com	305-129-4593
30	Sam	Moore	Accounting	sm@gmail.com	

Table_1 – Entities Relations Characteristics

Table 2 below shows three columns: the buyer's name, the **SKU** (stock keeping unit), and the names of the buyer's college major. The Buyers_Name may be associated with more than one SKU, and they can have multiple Departments.

Table_2 – Buyer and SKU_Numbers

	Buyers_Name	SKU_Numbers	Department
1	Judy Griffin	Griffin	Payroll

2	Lily Murphy	Murphy	Security
3	Sam Moore	Moore	Accounting

In the relational model of Databases:

- A **candidate key** is an element that controls all the other columns in a relation. The SKU_DATA relation has two candidate keys: SKU and SKU_Description. Buyer is an element, but it is not a candidate key because it controls only Department.

- **A primary key** is a specific choice of a minimal set of attributes (Columns) that is a single row of a table in a database relation. A table has only one primary key. The primary key can have one column, or it can be a composite. The definition of a table is given as **(SKU, SKU_Description, Department, Buyer)**, where **SKU** is the primary key of SKU_DATA and the (**OrdrNumber, SKU**) is the primary key of ORDER_ITEM.

SKU_DATA (SKU, SKU_Description, Department, Buyer)
ORDER_ITEM (OrderNumber, *SKU*, Quantity, Price, ExtendedPrice)

- A **surrogate key** is an artificial column that is added to a table to serve as the primary key when the primary key is large. RENTAL_PROPERTY is the relationship below.

RENTAL_PROPERTY (Street, City, State/Province, ZIP/PostalCode, Country, Rental_Rate)

- A **foreign key** is a column that is the primary key of a table. The term occurs because it is a key of a **table *foreign*** to the one in which it appears **as the primary key**. In the below two tables, COMPANY.CompanyName is the primary key of COMPANY, and EMPLOYEE.Company is a foreign key. In this text, we will show ***foreign keys in italics***:

COMPANY (CompanyName, BudgetCode, OfficeNumber, CompanyPhone)
EMPLOYEE (EmployeeNumber, LastName, FirstName, *Company*)

- **Anomalies** are problems that can occur in poorly planned database systems, unnormalised **databases** where the majority of the data are stored in one table **(a flat-file database)**. An **Insertion Anomaly** may be such that it is not possible to add a required piece of data unless another piece is unavailable data that is removed. When we delete one row, the structure of this table forces to lose facts about two different things: a machine and a repair. This condition is called a **deletion anomaly**.

3.3 Define first (1NF), second (2NF), and third normal (3NF) form

Normalization is a technique in database design used to organize tables in a way that reduces redundancy and dependency of data. The use of this technique splits first into a larger table then into smaller tables, and links all by using entity relationships. The designer of the relational model, Edgar Codd, came up with this theory of normalization that introduces the First Normal (1NF) form and then continues to extend the theory into a second (2NF) and third normal form (3NF). The main idea with this theory is that a table is about a specific topic and only supporting topics are included, which minimizes duplicate data, avoids data modification issues, and simplifies queries.

The **first normal form (1NF)**, like all normal forms, has specific requirements to be validated as 1NF. A table, as below, must be two dimensional with rows and columns, and each row must contain data that relates to something. Each column must contain data for a single attribute of the thing it's describing, each cell of the table must have only a single value, entries in any column must have the same type of data (Ex. If the entry in one row of a column contains hair color, all the other rows must contain hair color as well), each column must have a unique name, all rows should be uniquely identified (has to have some unique ID), and the order of the columns and rows must have no significance. Tables in first normal form are subject to deletion and insertion anomalies; they may prove useful in some applications but can be unreliable in others.

Customer_ID	Customer Name	Item	Price	Supplier	Supplier Phone
allen_d	Allen Davidson	Xbox One	250	Microsoft	1-800-BUY-XBOX
cameron123	Cameron Cherry	PlayStation 4	300	Sony	1-800-BUY-SONY
trinity9	Trinity Wilson	PS Vita	200	Sony	1-800-BUY-SONY

Figure 1: This is an example of a table in the 1st Normal Form.

Second normal form (2NF) must have all attributes or non-key columns dependent on the key. For example, if the data is based on making an order as shown in Figure 1, the price of the item isn't determined by the primary key or unique identifier of the customer, so this would break the second normal form. In this case, the table can be split into two, with one table having the primary key as Customer_ID and all of their personal information and the second being a table with the primary key as the item name and the supplier information and prices. All of this information in each table is now correctly dependent on the primary key.

Customer_ID	Customer Name
allen_d	Allen Davidson
cameron123	Cameron Cherry
trinity9	Trinity Wilson

Item	Supplier	Supplier Phone	Price
Xbox One	Microsoft	1-800-BUY-XBOX	250
PlayStation 4	Sony	1-800-BUY-SONY	300
PS Vita	Sony	1-800-BUY-SONY	200

Figure 2: Representation of the 2nd Normal Form.

Since the second normal form has two separate tables, a junction table is necessary where both keys are represented in the same table to show who bought what. This can also be used as a compound key where two primary keys are conjoined to represent each other.

Customer_ID	Item
allen_d	Xbox One
cameron123	PlayStation 4
trinity9	PS Vita

Figure 3: Junction Table

In the **third normal form (3NF),** all columns can be determined only by the key in the table and no other column. Having this type of normalization gets rid of redundancy and is especially vulnerable to some types of modification anomalies. Take a look at Figure 2 where the items table has duplicate supplier phone numbers and supplier names. If Sony were to have more

products under their item key, the supplier phone number would keep repeating because it's directly related to the supplier name and will always be that same phone number.

Item	Supplier	Supplier Phone	Price
Xbox One	Microsoft	1-800-BUY-XBOX	250
PlayStation 4	Sony	1-800-BUY-SONY	300
PS Vita	Sony	1-800-BUY-SONY	200

Separating these two tables into individual representations makes all of the columns dependent on only the key in the table and no other column, making the third normal form valid.

Item	Supplier	Price
Xbox One	Microsoft	250
PlayStation 4	Sony	300
PS Vita	Sony	200

Supplier	Supplier Phone
Microsoft	1-800-BUY-XBOX
Sony	1-800-BUY-SONY

- Some researchers discovered that anomalies could occur, and the conditions for **Boyce-Codd Normal Form (BCNF)** were identified. These normal forms are **well-defined so the relation in BCNF is in 3NF**, a relation in 3NF is in 2NF, and a relation in 2NF is in 1NF can occur.
- Therefore, the normal forms 2NF through BCNF concern anomalies that happen from functional dependencies. They led to the definition of **fourth normal form (4NF)** and **fifth normal form (5NF)**.
- Table 3 shows other anomalies which occurred because of another kind of dependency called a **multivalued dependency.** Those anomalies could be eliminated by placing each multivalued dependency in a relation of its own, a condition known as 4NF.

Table_3 – Form of Normalizations Theory

Anomaly	Norma Forms	Design Principles
Functional Dependencies	1NF, 2NF, 3NF, BCNF	BCNF: Design tables so that every determinant is a candidate key.
Multivalued Dependencies	4NF	4NF: Move each multivalued dependency to a table of its own.
Data Constrained	5NF, DK/NF	DK/NF: Make every constraint a logical consequence of candidate keys and domains.

The **fifth normal form (5NF),** also known as **Project-Join Normal Form (PJ/NF)** can lead to an anomaly where a table can be **split apart but not correctly joined back together**.

However, the conditions this condition is rather complex, and generally, if a relation is in 4NF it is in 5NF. For more information about 5NF, consult the Wikipedia article at https://en.wikipedia.org/wiki/Fifth_normal_form.

3.4 State Two Properties of Candidate Keys

A candidate key is a conglomerate of attributes that identify a database record in a unique way without referencing any other key data from the database. A table may contain one or more candidates and one of those candidate keys has to be referred to as the primary key. The absolute requirement for a table to have is the primary key, but the maximum number of **candidate keys** is unlimited by any constraints.

The naming of a candidate key is coming from a super key with no redundant attribute and they are respectively selected from the set of super keys. Another name for **candidate key** could be **minimal super key.**

All candidate keys have common properties. One of the most important candidate key properties is that the attribute that is used for identification must remain the same for its lifetime. Another basic but important property of candidate keys is that the value of the attribute cannot be a null value. It's a form of identification; therefore, if the value has a null value, there is no way a query can be applied to this row and find the desired information. Each candidate key must contain minimum fields to ensure uniqueness.

StudID	Roll No	FirstName	LastName	Email
1	11	Ben	RandomA	abc@gmail.com
2	12	David	RandomB	xyz@gmail.com
3	13	Cameron	RandomC	mnoprq@gmail.com

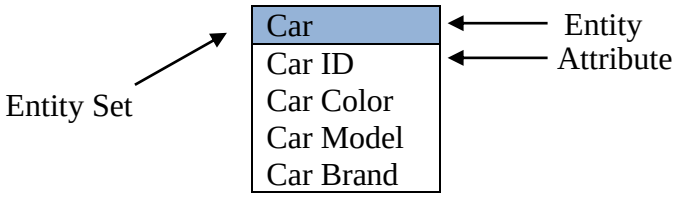
This example identifies each property in a unique and standard way. The studID, Roll Number and Email are unique attributes, therefore all three of them are candidate keys. Most importantly studID is a candidate key and also primary key. To uniquely identify each student, a query can use the **primary key**, which also serves as a candidate key. One of the other options would be to identify it by Roll Number or by the email, as only one email and Roll Number can exist.

Other examples of important candidate keys : Social Security Number, International Standard Book Numbers, Bank account numbers, Serial Numbers, Driver License Numbers, National Provider ID, Phone numbers and so on.

3.5 Section Concise Summary

An Entity Relationship (ER) model is a diagram that gives a good overview of the design of a database system with the aim of making it easier to be understood at a technical level. This model can be mapped to a relational schema which shows the relationship between its members. An ER diagram consists of multiple components: the entity, entity type, entity set, and attributes.

The entity of the ER diagram is an object or concept about which you want to store information. This component is the overall subject of the entity set, whilst the entity set is within the entity which holds the attributes of the entity's type. For example, in an entity called "Car", an entity set would consist of attributes "Car ID", "Car color", "Car model" and "Car brand". To explain this better, here is a visual representation.



The next important step in understanding ER mapping is attributes. These components are properties which define the entity type. There are many examples to describe this, including Name, DOB, Address, etc. The attribute that uniquely identifies each entity is called a **key attribute**, usually identifying each entity with a unique number. In the example above, Car ID would be the key attribute which can also be called "Primary Key" or PK. Car ID is the key attribute because every Car ID is unique and can identify each entity separately without duplicates. For more complex designs, there are multiple types of attributes: Composite attributes (compose of two or many combined attributes), Multivalued Attributes (consists of more than one value) and Derived attributes (attributes derived from other attributes).

A weak entity is an entity type for which a key attribute can't be defined. A company can store information of dependents (Parents, Children, Spouse) of an Employee but the dependents don't exist without employee.

A relational model for database management system is an approach to managing data using a table structure that is consistent with specific logic and properties. In database systems, there exist relations in a two-dimensional table of data. This typically has multiple named columns and an unknown quantity of rows for data to be entered. There are specific properties that define relations so that they can be easily identified by a user. There are six properties of relations:

1. Each relation (or table) in a database has a unique name.
2. Each entry at the intersection of each row and column is atomic (or single valued).
(There can be only one value associated with each attribute on a specific row of a table; no multivalued attributes are allowed in a relation.)
3. Each row is unique; no two rows in a relation can be identical.
4. Each attribute (or column) within a table has a unique name.
5. The sequence of columns (left to right) is insignificant. The order of the columns in a relation can be changed without changing the meaning or use of the relation.
6. The sequence of rows (top to bottom) is insignificant. As with columns, the order of the rows of a relation may be changed or stored in any sequence.

As stated previously, these relational tables use a certain structure that is consistent with specific logic and properties: this is called **normalization**. This process is a technique in database design that is used to organize tables in a manner that reduces redundancy and dependency of data. The use of this technique divides larger tables into smaller tables and links them using relationships. The creator of the relational model, Edgar Codd, came up with the theory of normalization with the introduction of the First Normal Form and then continued to extend this theory into a second and third normal form.

The first normal form, like all normal forms, has specific requirements to be validated as first normal form. A table in first normal form must be two dimensional with rows and columns, each row must contain data that pertains to something, each column must contain data for a single attribute of the thing it's describing, each cell of the table must have only a single value, entries in any column has to have the same type of data (Ex. If the entry in one row of a column contains hair color, all the other rows must contain hair color as well), each column must have a unique name, all rows should be uniquely identified (has to have some unique ID), and the order of the

columns and rows has no significance. Tables in first normal form are subject to deletion and insertion anomalies and may prove useful in some applications but can be unreliable in others.

Customer_ID	Customer Name	Item	Price	Supplier	Supplier Phone
allen_d	Allen Davidson	Xbox One	250	Microsoft	1-800-BUY-XBOX
cameron123	Cameron Cherry	PlayStation 4	300	Sony	1-800-BUY-SONY
trinity9	Trinity Wilson	PS Vita	200	Sony	1-800-BUY-SONY

Figure 1: This is an example of a table in the 1st Normal Form.

Second Normal Form
Must have all attributes or non-key columns dependent on the key.

Customer_ID	Customer Name
allen_d	Allen Davidson
cameron123	Cameron Cherry
trinity9	Trinity Wilson

Item	Supplier	Supplier Phone	Price
Xbox One	Microsoft	1-800-BUY-XBOX	250
PlayStation 4	Sony	1-800-BUY-SONY	300
PS Vita	Sony	1-800-BUY-SONY	200

Figure 2: Representation of the 2nd Normal Form.

Third Normal Form
All columns can be determined only by the key in the table and no other column.

Customer_ID	Customer Name
allen_d	Allen Davidson
cameron123	Cameron Cherry
trinity9	Trinity Wilson

Supplier	Supplier Phone
Microsoft	1-800-BUY-XBOX
Sony	1-800-BUY-SONY

Item	Supplier	Price
Xbox One	Microsoft	250
PlayStation 4	Sony	300
PS Vita	Sony	200

Having this type of normalization gets rid of redundancy and is especially vulnerable to some types of modification anomalies. If Sony were to have more products under their item key, the supplier phone number would keep repeating because it's directly related to the supplier name and will always be that same phone number. Separating these two tables into individual representations makes all of the columns dependent on only the key in the table and no other column, making the third normal form valid.

Candidate keys are a collective of attributes that identify a database record in a unique way without referencing any other key data from the database. A table may contain one or more candidates and one of those candidate keys has to be referred to as the primary key. The absolute requirement for a table to have is the primary key, but the maximum number of candidate keys is unlimited by any constraints.

3.6 Extended Resources

1. A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It's the database designers who design the schema to help programmers understand the database and make it useful.

https://www.tutorialspoint.com/dbms/dbms_data_schemas.htm

---

2. This video shows converting ER Diagrams to Schemas | SQL - Discuss the role of designing databases in the analysis and design of an information system. Learn how to transform an entity-relationship (ER) Diagram into an equivalent set of well-structured relations

https://www.youtube.com/watch?v=xQRRf5fOAt8

---

3. Normalization? 1NF, 2NF, 3NF, BCNF Database Example - The inventor of the relational model Edgar Codd proposed the theory of normalization with the introduction of the First Normal Form, and he continued to extend theory with Second and Third Normal Form. Later he joined Raymond F. Boyce to develop the theory of Boyce-Codd Normal Form.

https://www.guru99.com/database-normalization.html

---

4. Video on Normalizations has three steps: 1nf, 2nf, and 3nf. These stand for first normal form, second normal form, and third normal form. Each step has rules on what is allowed or not allowed in our database design. Each normal form gets progressively more strict.

https://www.youtube.com/watch?v=TOwhX0IhLD8

---

5. Video on Multivalued dependencies According to database theory, a multivalued dependency is a full constraint between two sets of attributes in a relation. In contrast to the functional dependency, the multivalued dependency requires that certain tuples be present in a relation. Wiki: https://en.wikipedia.org/wiki/Multivalued_dependency

Part_1 https://www.youtube.com/watch?v=A6VGW1_1UHA

Part_2 https://www.youtube.com/watch?v=07iWNlfmLcg&t=25s

References

Database Normalization (Explained in Simple English). (2020, February 19). Retrieved from https://www.essentialsql.com/get-ready-to-learn-sql-database-normalization-explained-in-simple-english/

Taylor, A. G. (n.d.). SQL First, Second and Third Normal Forms. Retrieved from https://www.dummies.com/programming/sql/sql-first-second-and-third-normal-forms/

Normalization - 1NF, 2NF, 3NF and 4NF. (2015). Retrieved from https://www.youtube.com/watch?v=UrYLYV7WSHM&t=

Mapping from ER Model to relational Model (2020, February 20). Retrieved from https://www.geeksforgeeks.org/mapping-from-er-model-to-relational-model

Introduction of ER Model (2020, February 20) Retrieved from https://www.geeksforgeeks.org/introduction-of-er-model/

Hoffer, Jeffrey A, V Ramesh, and Heikki Topi. *Modern Database Management*. , 2011. Print.