

DATABASE SYSTEM

CSC 472



4/1/2021

Table of Contents

[Table of Figures ii](#_Toc68367584)

[Online Food Ordering System 4](#_Toc68367585)

[ER Diagram of Online Food Ordering System 7](#_Toc68367586)

[ER-Diagram Explanation 9](#_Toc68367587)

[Schema Diagram of Online Food Ordering System 10](#_Toc68367588)

[Steps involved in setting up a database for the operation of any organisation 12](#_Toc68367589)

[Setting up a good database is therefore is one that: 12](#_Toc68367590)

[Data Abstraction 13](#_Toc68367591)

# Table of Figures

[Figure 1\_online\_food\_ordering\_system\_erd](../../../../../../../../C:/Users/USER/Desktop/CSC_ASS/CSC_472_II_Adefemi_Micheal_Afuwape_(206323).docx" \l "_Toc68140195) 8

[Figure 2\_online\_food\_ordering\_system\_schema\_level\_diagram](../../../../../../../../C:/Users/USER/Desktop/CSC_ASS/CSC_472_II_Adefemi_Micheal_Afuwape_(206323).docx" \l "_Toc68140196) 11

**ASSIGNMENT II**

COURSE CODE: **CSC472**

COURSE TITLE: **DATABASE SYSTEMS**

**Question**

1. Demonstration of understanding of abstraction and schema levels
2. As a Database Administrator (DBA), briefly explains in details the steps involved in setting up a database for the operation of any organisation of your choice. Demonstrate richly your understanding of the 3 levels of abstraction and schema.

**Answer**

* For the organization I will be using an **Online** **Food Ordering System** as an example and I will be listing out all known entities and attributes as well as show the relationship between them. Furthermore, some schema diagrams will be added and abstraction level will be pointed out.

# Online Food Ordering System

In an Online Food Ordering System, some of the entities there are

1. User
2. Payment
3. Order
4. Order details
5. Customer
6. Menu
7. Menu type
8. Rating
9. Web Site information

I will be taking each entity and writing about their attributes

1. **Admin**: The user entity will have the following attributes:
   1. Admin ID – which will be the **primary key**
   2. Full Name – which is a **composite attribute**
   3. User Contact – which is a **composite attribute**
   4. Username
   5. Password
2. **Payment**: The payment entity will have the following attributes
   1. Payment ID – which is the **primary key**
   2. Order ID – which is a **foreign key**
   3. Processed By – which is a **foreign key**
   4. Amount
   5. Paid By
   6. Payment Date
3. **Order**: The order entity will have the following attributes
   1. Order ID – which is a **primary key**
   2. Customer ID – which is a **foreign key**
   3. Processed By – which is a **foreign key**
   4. Order Date
   5. Total Amount
   6. Order Status
4. **Order Details**: The order details entity will have the following attributes
   1. Order Details ID – which will be a **primary key**
   2. Order ID – A **foreign key**
   3. Menu ID – A **foreign key**
   4. Amount
   5. No of orders
   6. Total Amount
5. **Customer**: The customer entity will have the following attributes
   1. Customer ID – which will be the **primary key**
   2. Customer Full Name - which is a **composite attribute**
   3. Customer Contact - which is a **composite attribute**
   4. Customer Username
   5. Customer Password
   6. Customer Account Status
6. **Menu**: The menu entity will contain the following attributes
   1. Menu ID – which will be the **primary key**
   2. Menu Type ID – **foreign key**
   3. Menu Name
   4. Menu Price
   5. Menu Image
   6. Menu Ingredients
   7. Menu Status
7. **Menu** Type: The menu type entity will contain the following attributes
   1. Menu Type ID – which will be a **primary key**
   2. Menu Type Name
   3. Description
8. **Rating**: The rating entity will have the following attributes
   1. Rating ID – which will be a **primary key**
   2. Menu ID – **foreign key**
   3. Customer ID – **foreign key**
   4. Rating Score
   5. Remarks
   6. Date Recorded
9. Web Site Information

**Component Attribute**: This are types of attributes that can be divided into sub-parts (other attributes) an example is the Full Name which can be further divided into first name, last name and middle name. Composite attributes help us to group together related attributes, making the modeling cleaner

Before drawing the ER Diagram and Schema Level Diagram I will point out how this entity relates with each other

* The user manage/update the web site information and this is a **one-to-one relationship**
* The user also processes orders coming from the customers and this is a **one-to-many relationship**
* The user process payments coming from the customers, this is also a **one-to-many relationship**
* The customer places orders, this is a **one-to-many relationship**
* The order information coming from the customer can consist of one or more items and therefore is a **one-to-many relationship**
* The order detail can consist of one or more menu and its also a **one-to-many relationship**
* Order will be connected to payment since each other will have different payment, this therefore means a **one-to-one relationship**
* The customer can also give rating on a particular menu which is a **one-to-one relationship**
* A specific menu can have more than one rating from different customers, this is a **one-to-many relationship**
* A menu can belong to one specific menu type this is a **one-to-one relationship**

# ER Diagram of Online Food Ordering System

ER Diagram is a visualization of the relationships between tables in a database. It contains, at the very least, table names represented as squares linked by lines representing primary and international main constraints. It also includes column names and symbols, as well as details about the relationship between the columns (one-to-one, one-to-many, many-to-many. ER Diagrams are usually drawn in three levels, and although all three levels of an ER model contain entities with attributes and relationships, the purposes for which they are generated and the audiences for which they are intended to reach vary. The business analyst uses a conceptual and logical model to model the business objects that exist in the system, while the database designer or database engineer elaborates the conceptual and logical ER model to create the physical model, which introduces the physical database structure ready for database development. The three level diagrams are:

* Conceptual ER Diagram: Conceptual ERD models the **business objects that should exist in a system and the relationships between them**. By recognizing the business objects involved, a conceptual model is created to present an overall image of the system. It determines which entities are present, not which tables are present.
* Logical ER Diagram: Logical ERD is a **detailed version of a Conceptual ERD**. By specifically specifying the columns in each entity and adding operational and transactional entities, a logical ER model is generated to enrich a conceptual model. While a logical data model is independent of the database system in which the database would be built, it can also be taken into account if it has an effect on the design.
* Physical ER Diagram: Physical ERD represents the **actual design blueprint of a relational database**. A physical data model adds to the logical data model by defining the form, length, and nullability of each column. Since a physical ERD represents how data should be organized and linked in a particular DBMS, it's critical to think about the conventions and constraints of the database system where the database will be generated. Ensure that the DBMS supports the column types and that no reserved terms are used in the naming of entities and columns.

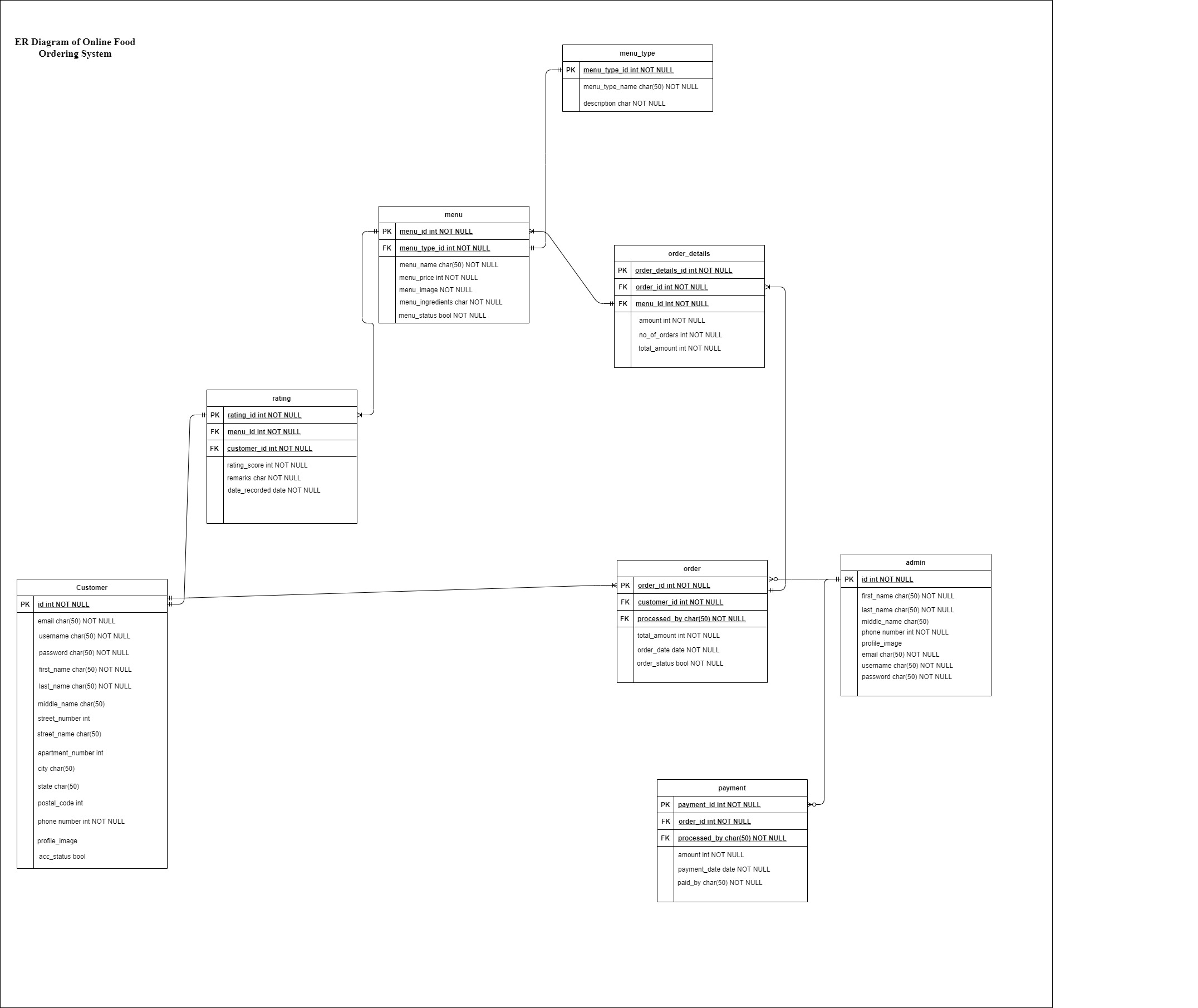
****

Figure 1\_online\_food\_ordering\_system\_erd

# ER-Diagram Explanation

For the ER diagram above the explanation is as follow:

1. One customer can place many orders
2. One customer can submit one rating for a meal
3. Many order details have a single order
4. An admin can process many order
5. Many payments can be processed by one admin
6. A menu belongs to one menu type
7. A menu can have one or more rating
8. Also, one or more menu can only have one order details

# Schema Diagram of Online Food Ordering System

The database schema is a definition of the database's actual creation. It's a broad term that refers to tables, columns, causes, relationships, key constraints, roles, and procedures as a whole. It may refer to either a text that defines anything (such as an XML Schema) or an abstraction of the database's structure ("It would be difficult to change the schema of the database at this point"). It has nothing to do with the data or the rows that have been added to the schema. You would **insert** data into an existing schema. The Database Schema can then be represented using **Schema Diagrams**.

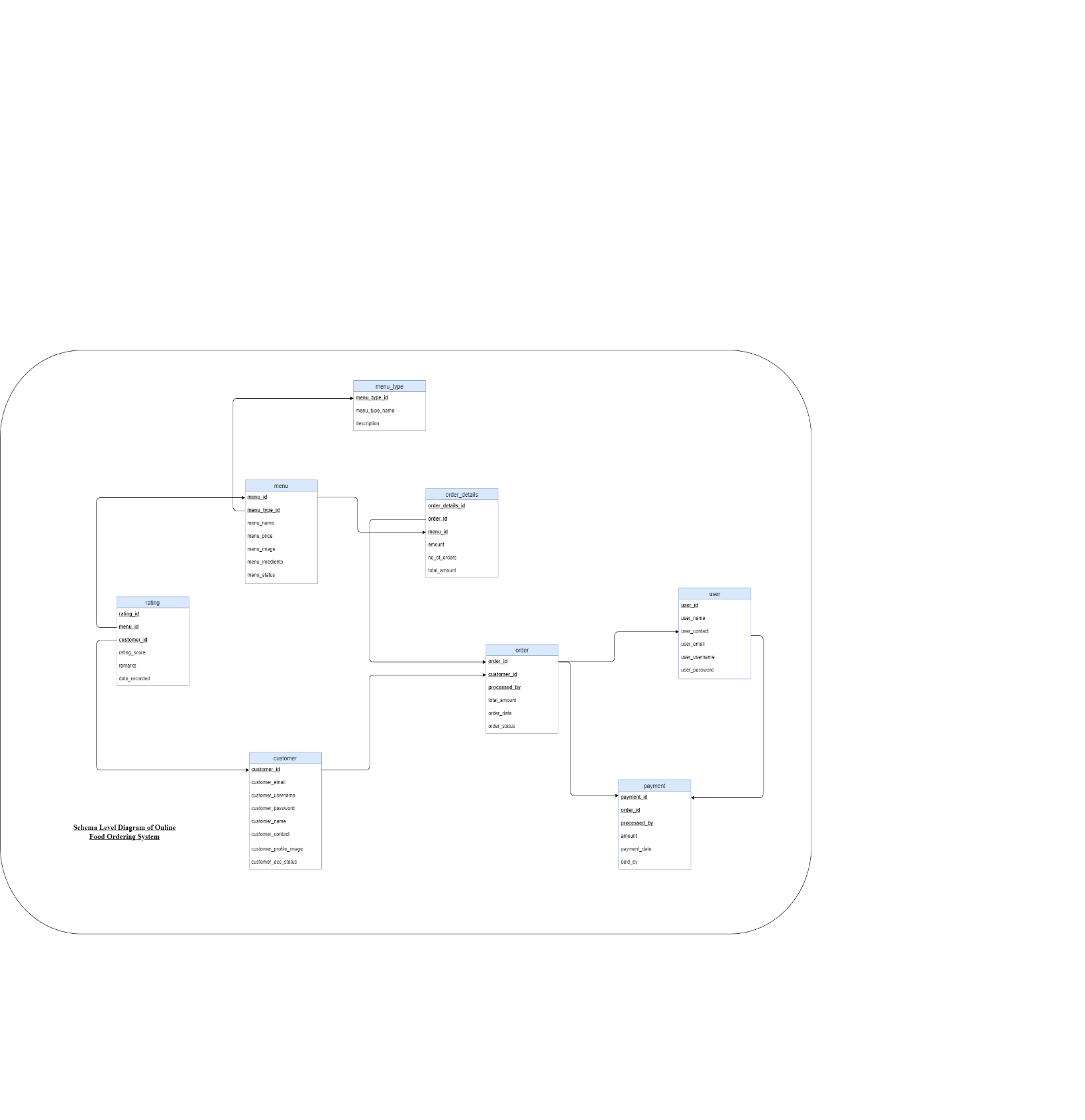


Figure 2\_online\_food\_ordering\_system\_schema\_level\_diagram

# Steps involved in setting up a database for the operation of any organisation

* **Determine the purpose of your database**: This helps prepare you for the remaining steps.
* **Find and organize the information required:** Gather all of the types of information you might want to record in the database, such as product name and order number.
* **Divide the information into tables:** Divide your information items into major entities or subjects, such as Products or Orders. Each subject then becomes a table.
* **Turn information items into columns**: Decide what information you want to store in each table. Each item becomes a field, and is displayed as a column in the table. For example, an Employees table might include fields such as Last Name and Hire Date.
* **Specify primary keys:** Choose each table’s primary key. The primary key is a column that is used to uniquely identify each row. An example might be Product ID or Order ID.
* **Set up the table relationships:** Look at each table and decide how the data in one table is related to the data in other tables. Add fields to tables or create new tables to clarify the relationships, as necessary.
* **Refine your design:** Analyze your design for errors. Create the tables and add a few records of sample data. See if you can get the results you want from your tables. Make adjustments to the design, as needed.
* **Apply the normalization rules:** Apply the data normalization rules to see if your tables are structured correctly. Make adjustments to the tables, as needed.

## Setting up a good database is therefore is one that:

* Divides your information into subject-based tables to reduce redundant data.
* Provides Access with the information it requires to join the information in the tables together as needed.
* Helps support and ensure the accuracy and integrity of your information.
* Accommodates your data processing and reporting needs.

# Data Abstraction

The three levels of data abstraction are:

* + Physical level of data abstraction
  + Logical level of data abstraction
  + View level of data abstraction

1. Physical Level of Data Abstraction: The physical level of data abstraction is also known as INTERNAL LEVEL. The physical level is the lowest abstraction level of all. It explains how data in the database is stored. This level defines implementation specifics such as indexing methods such as B+ trees or hashing, as well as access methods such as sequential or random access. Essentially, we map the conceptual level to the characteristics and constraints of the chosen models. As a result, the DBMS is no longer based on the internal model. Assume we select relational database management systems; our conceptual models will then be mapped to the internal level. The ER model's entities will be mapped to tables. From the Online Food Ordering System, some of the physical level of data abstraction are:
2. Logical Level of Data Abstraction: The logical level of data abstraction is also known as the conceptual level. We can see what data is stored in the database at this stage without knowing about the implementation specifics including data structures and tree implementations. This level also offers details on the relationships between the various fields and database tables. This association can be many-to-many, one-to-many, many-to-one, or some of the other joins. We can see an example of the data flow within the company here. This level of data is only available to database managers.

Taking the example of an Online Food Ordering System, the relationship between an entity ‘Customer and another entity Order, can be one-to-many. The fields describing the entity customer would be the same as describing a general entity Person in along with fields such as first name, last name, address, etc. Any changes to this level should not affect the physical or view level. For example, say we are to add field Favorite to the entity Customer, it should neither change the way a user is able to access data nor should it affect the way data is stored in the database.

1. View Level of Data Abstraction: The view level of data abstraction is also known as EXTERNAL LEVEL. Any organization's database protection and credibility are important. For the company to run smoothly, the right people must have access to the right data. This necessitates the use of view level abstraction. From the three levels of data abstraction, this is the highest level. Only the data that is important to the users is available in this way.

To understand the view level better, lets use the Online Food Delivery System as example, the users will be customer, at the view level, a customer will not have access to the details of other customers. The admin might not have full access to customer card information but will surely have access to basic customer information like first name, username etc.