Computer Aided Software Engineering (CASE) Tools /lec3

Marrwa alabajee

Contents

Classification of CASE Systems

- A number of risks are inherent whenever we attempt to categorize CASE tools. There is a subtle implication that to create an effective CASE environment, one must implement all categories of tools this is simply not true.

Confusion (or antagonism) can be created by placing a specific tool within one category when others might believe it belongs in another category.

- Some readers may feel that an entire category has been omitted there by eliminating an entire set of tools for inclusion in the overall CASE environment. In addition, simple categorization tends to be flat that is, we do not show the hierarchical interaction of tools or the relationships among them.

Classification of CASE Systems

- But even with these risks, it is necessary to create a taxonomy of CASE tools to better understand the breadth of CASE and to better appreciate where such tools can be applied in the software engineering process.
- CASE tools can be classified by **function**, by **their role as instruments for managers or technical people**, by **their use in the various steps of the software engineering process**, by **the used type (or kind) of modeling** ,by **the environment architecture (hardware and software) that supports them**, or even by **their origin** or **cost** .

Upper and Lower CASE

The most popular classification of CASE technology and tools is based on the distinction made between the early and late stages of systems development.

Many of the current CASE tools deal with the management of the system specification only by supporting strategy, planning and the construction of the conceptual level of the enterprise model.

These tools are often termed upper CASE tools because they assist the designer only at the early stages of system development and ignore the actual implementation of the system.

Upper and Lower CASE

The emphasis in upper CASE is to describe the mission, objectives, strategies, operational plans, resources, component parts etc. of the enterprise and provide automated support for defining the logical level of the business, its information needs and designing information systems to meet these needs.

Upper CASE tools support traditional diagrammatic languages like Entity Relationship Diagrams, Data Flow Diagrams, Structure Charts etc. providing mainly draw, store as well as documentation facilities. They support a limited degree of verification, validation and integration of the system specifications due to the inherent lack of formal foundations for the requirements modelling formalisms.

Upper and Lower CASE

Other CASE tools deal with the application development itself with regard to the efficient generation of code. These are termed lower CASE tools because they assist the developer at the stage of system generation and ignore the early stages of system requirements specification.

The starting point of the system development with lower CASE tools is the conceptual model of the information system. The conceptual modelling formalism is usually, based on formal foundations in order to allow for automatic mapping to executable specifications and efficient validation and verification of the system specification itself.

Upper and Lower CASE

Lower CASE tools employ mapping algorithms to automatically transform formal specifications into an executable form. This includes, among others, transformation of specifications to relational database schemas, normalization of database relations and SQL code generation.

The majority of these tools facilitate rapid prototyping of specifications in terms of the functionality of the system. They do not support the development process itself but rather, they offer a powerful tool for making system design more effective and efficient.

Upper and Lower CASE

- ❖ Integrated CASE, or I-CASE, contains functionality found in both upper CASE and lower CASE tools. They are helpful in all the stages of SDLC, from Requirement gathering to Testing and documentation. In this architecture, the repository plays a more active role in that all tools can interface and exchange data with it.

- ❖ A repository, holds data fields and definitions and ensures that data integrity is maintained throughout the development lifecycle. As a consequence, ICASE allow tools to work together relatively seamlessly and alleviates much of the stop-start nature of non-ICASE environments. Figure 1.3 illustrate these categories of CASE tools.

Upper and Lower CASE

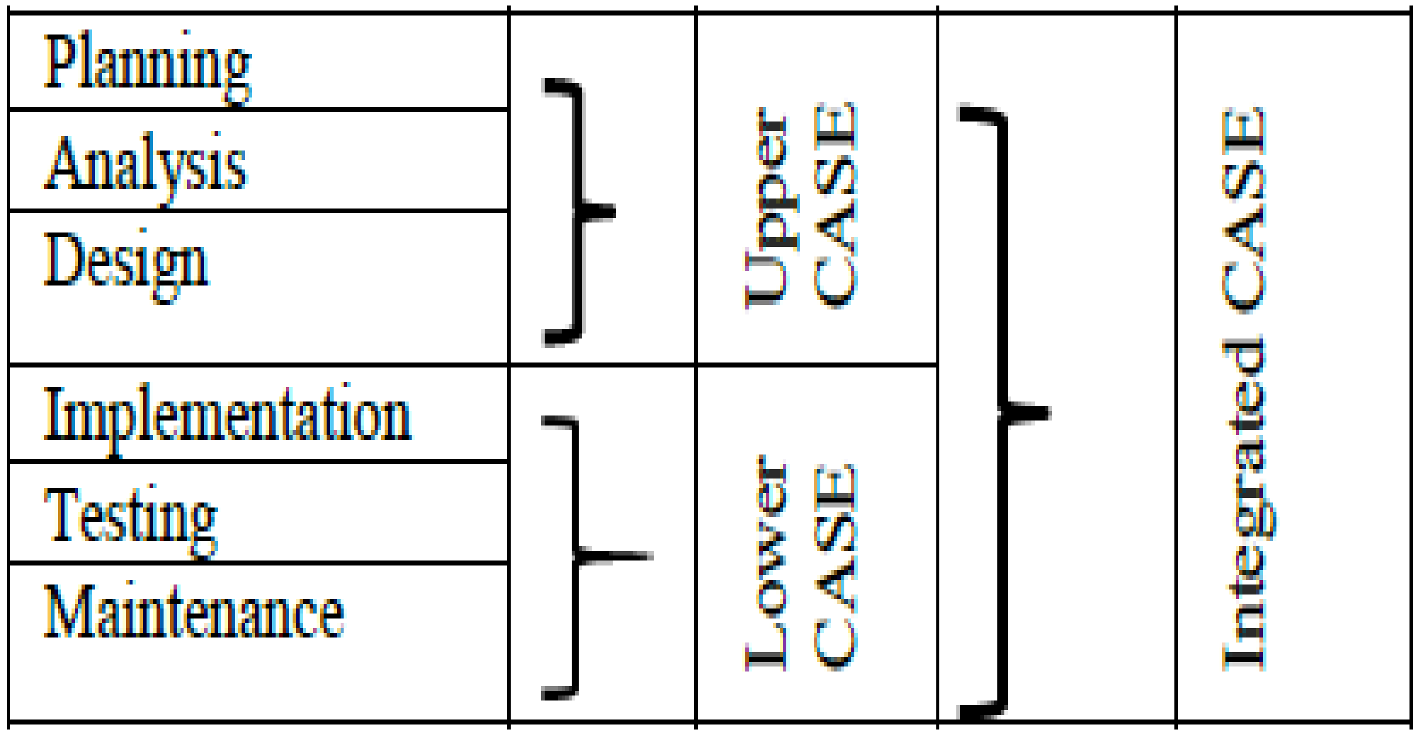


Figure 1.3 Categories of CASE tools

Fuggetta’s classification

Fuggetta’s classification is done on the basis of CASE-system categories and reflects the degree of integration on functionality. All observed systems are divided by him into three groups:

1. "**Tools**": support only specific tasks in the software process.
2. "**Workbenches**": support only one or a few activities.
3. "**Environments**": support (a large part of) the software process.

Fuggetta’s classification- Tools

Tools. A CASE tool is a software component supporting a specific task in the computer-aided operating organization and software engineering process:

1. **Editing tools (editors):** can be classified in two subclasses: textual editors and graphical editors.

text editor, a software program that allows users to create or manipulate plain text computer files. They are often used in the field of computer programming. for Examples **NotePad** and **WordPad** - Microsoft Windows included text editors.

Fuggetta’s classification- Tools

A graphical editor is a type of computer program that allows a user to edit and manipulate graphical images through a variety of methods. For example **Adobe Photoshop, Clip Studio Paint.**

2. **Programming tools:** these tools are used to support coding and code restructuring .The four main subclasses, singled out by us, are coding and debugging, code generators, code restructures, and code analyzers.

3. **Verification and validation tools**: this class includes tools that support program validation and verification.

Fuggetta's classification- Tools

Validation aims at ensuring that the product's functions are what the customer really wants .
while **verification** aims at ensuring that the product under construction meets the requirements definition.

4. **Configuration management tools**: Configuration Management (CM) techniques coordinate and control the construction of a system composed of many parts.
CM :is a systems engineering process for establishing consistency of a product's attributes throughout its life.
any future changes to these assets are known and tracked.its like up-to-date inventory for your technology assets.
5. **Project-management tools:** the three main subclasses, singled out by us, are execution of specific tasks management ,project management, portfolio of projects management.

Fuggetta’s classification- Workbenches

Workbenches: They integrate in a single application several tools supporting specific computer-aided operating organization and software engineering process activities:

1. **Business planning and modeling workbenches**: this class includes products to support the identification and description of a complex business.

2. **Analysis and design workbenches**: they automate most of the analysis and design methodologies.

3.User-interface development workbenches :these products do not help with specific software-process activities but rather with user-interface design and development.

Fuggetta’s classification- Workbenches

4. **Programming and designing of databases and files workbenches** : these workbenches provide integrated facilities supporting programming.

5. Verification and validation workbenches : this class of workbenches includes products that help with module and system testing.

6. Maintenance and reverse-engineering workbenches: these workbenches provide "forward" ,"reverse" development of system process.

7. Configuration-management workbenches: the workbenches in this class integrate tools supporting version control, configuration building, change control, registration of the state of configuration management objects, possibility of development of "client-server" applications of demanded configuration etc.

Fuggetta’s classification- Environments

Environments. The given group includes systems of the following kinds:

1. **Toolkits**: toolkits are loosely integrated collections of products easily extended by aggregating different tools and workbenches.
2. **Language-centered**: this environments are written in the language for which they were developed, thus letting users customize and extend the environment and reuse part of it in the applications under development.

3. **Integrated :** they operate using standard mechanisms so users can integrate tools and workbenches according :

Fuggetta’s classification- Environments

Platform integration

• Tools run on the same hardware/software platform

Data integration

• Tools operate using a shared data model

Presentation integration

• Tools offer a common user interface

4. **Fourth generation**: this is sets of tools and workbenches supporting the development of a specific class of program electronic data processing and business-oriented applications.

5. Process-centered : they are based on a formal definition of the software process.

Fuggetta’s classification

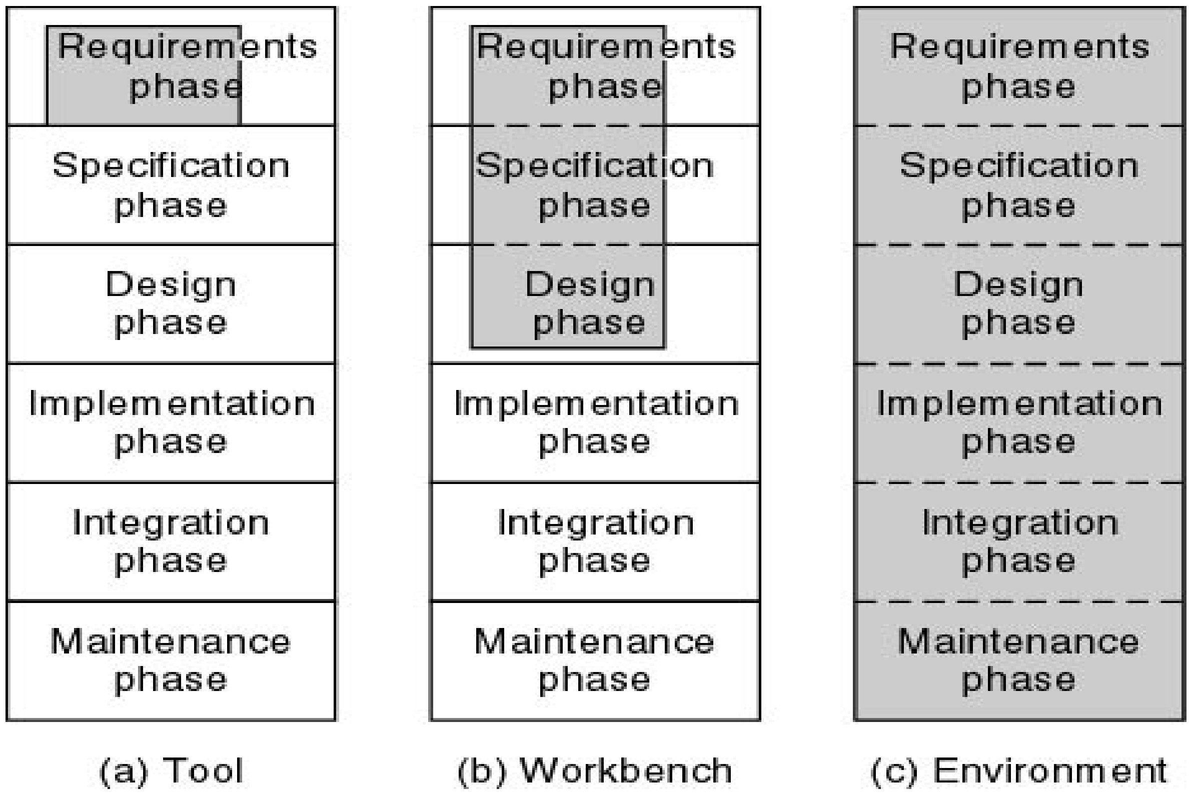


Figure 1.4 Fuggetta’s classification

Database Diagram Design Tools

A database schema is the blueprints of your database, it represents the description of a database structure, data types, and the constraints on the database. And designing database schemas is one of the very first and important steps to start developing any software.

1. **dbdiagram.io**
2. **Lucidchart**
3. **QuickDBD**
4. **ERD Plus**
5. DrawSQL
6. Miro

ERD Plus

ERD Plus is a basic database modeling tool for creating Entity Relationship Diagrams, Relational Schemas, Star Schemas, and SQL DDL statements.

- Automatically convert ER Diagrams into Relational Schemas
- Export SQL.
- Export diagrams as a PNG.
- Save diagrams safely on our server.

ERD Plus

- **ER Diagram** stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

- ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships.

ERD Plus

Following are the main components and its symbols in ER Diagrams:

Rectangles: This Entity Relationship Diagram symbol represents entity types

Ellipses : Symbol represent attributes

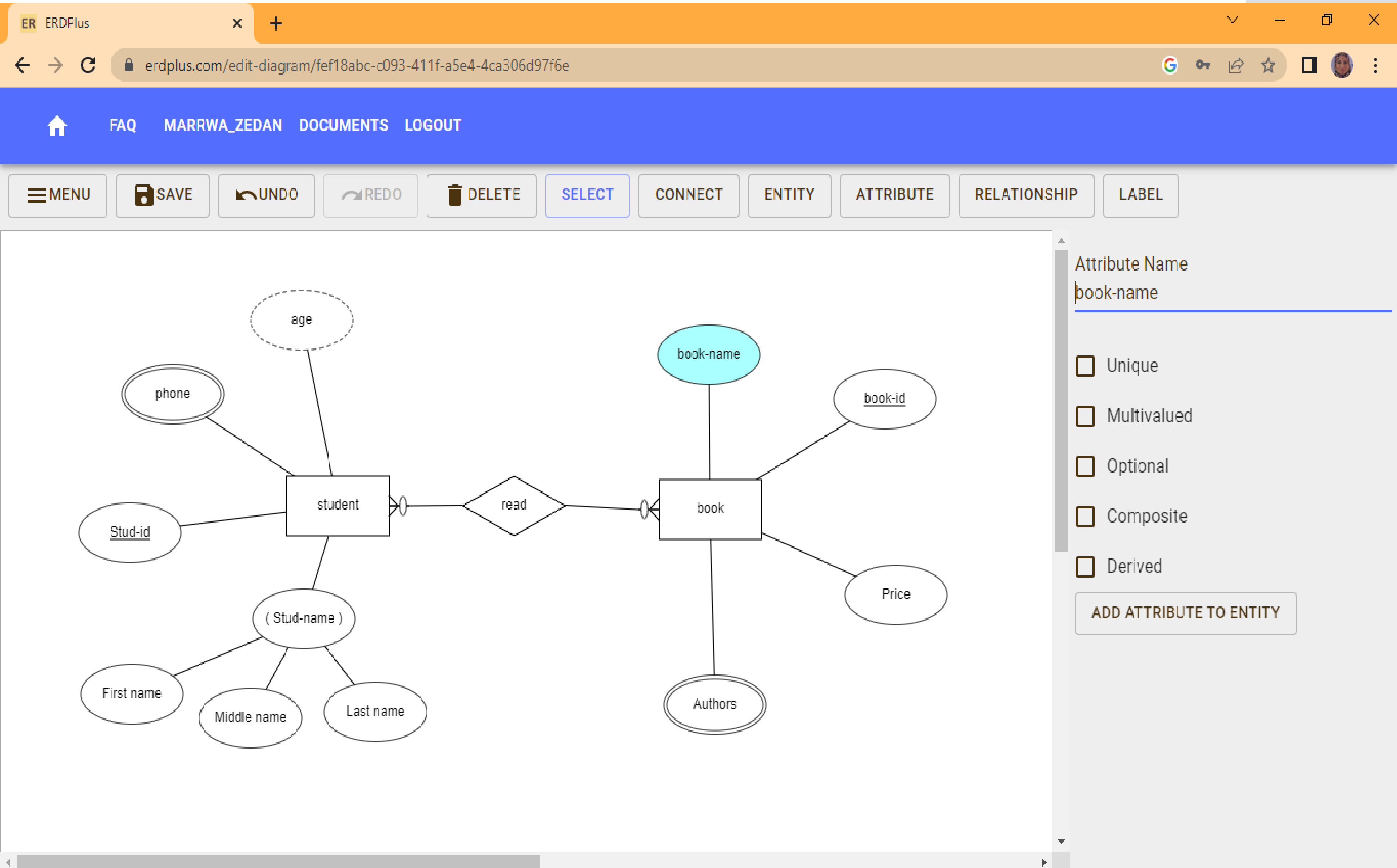
Diamonds: This symbol represents relationship types

Lines: It links attributes to entity types and entity types with other relationship types

Primary key: attributes are underlined

Double Ellipses: Represent multi-valued attributes

ERD Plus



Attribute Name
book-name

- [ ] Unique
- [ ] Multivalued
- [ ] Optional
- [ ] Composite
- [ ] Derived

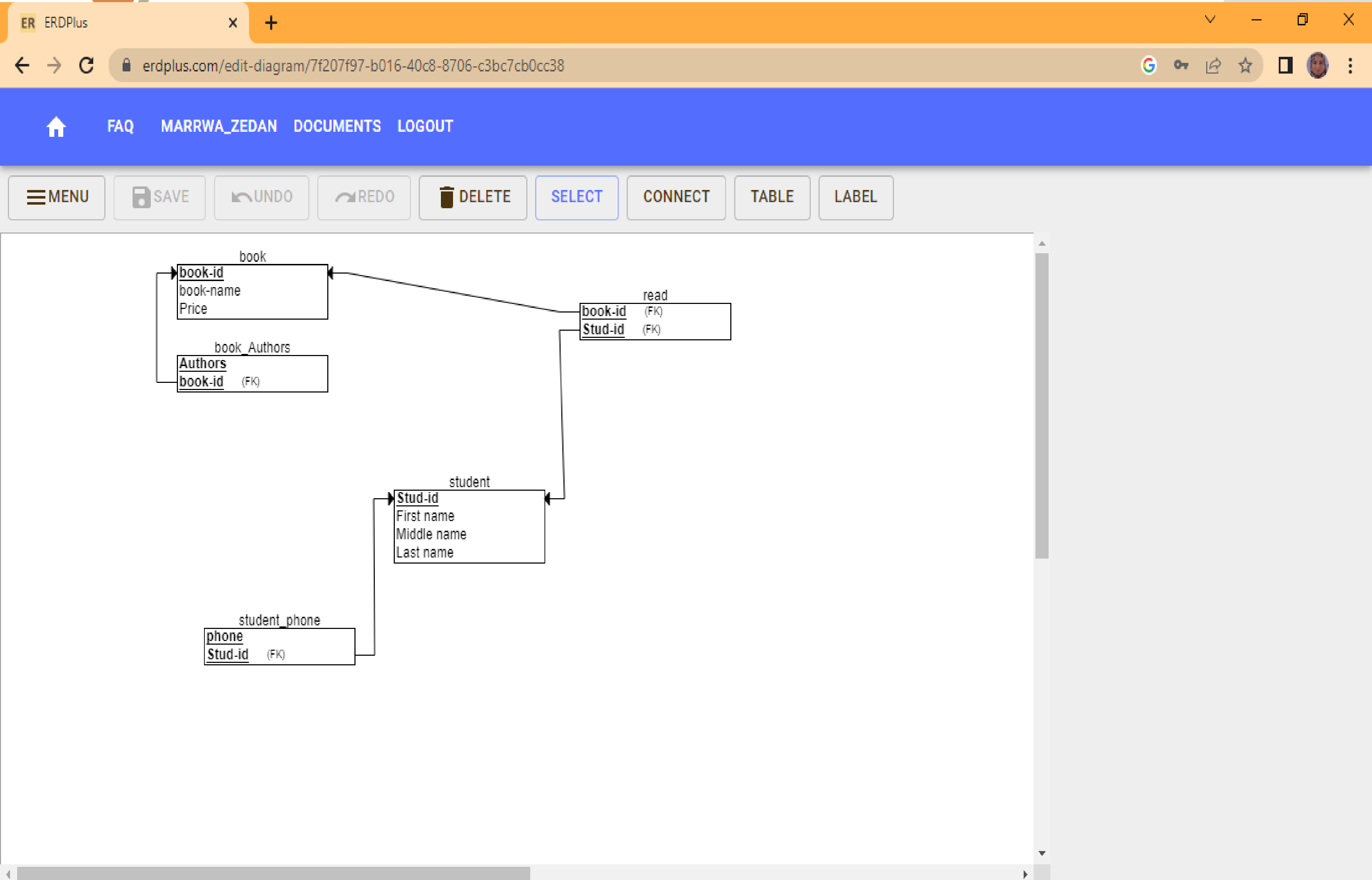
ERD Plus

What is a relational schema?

A relational schema is a blueprint used in database design to represent the data to be entered into the database and describe how that data is structured in tables (called relations in relational schemas). The schema describes how those tables relate to each other.

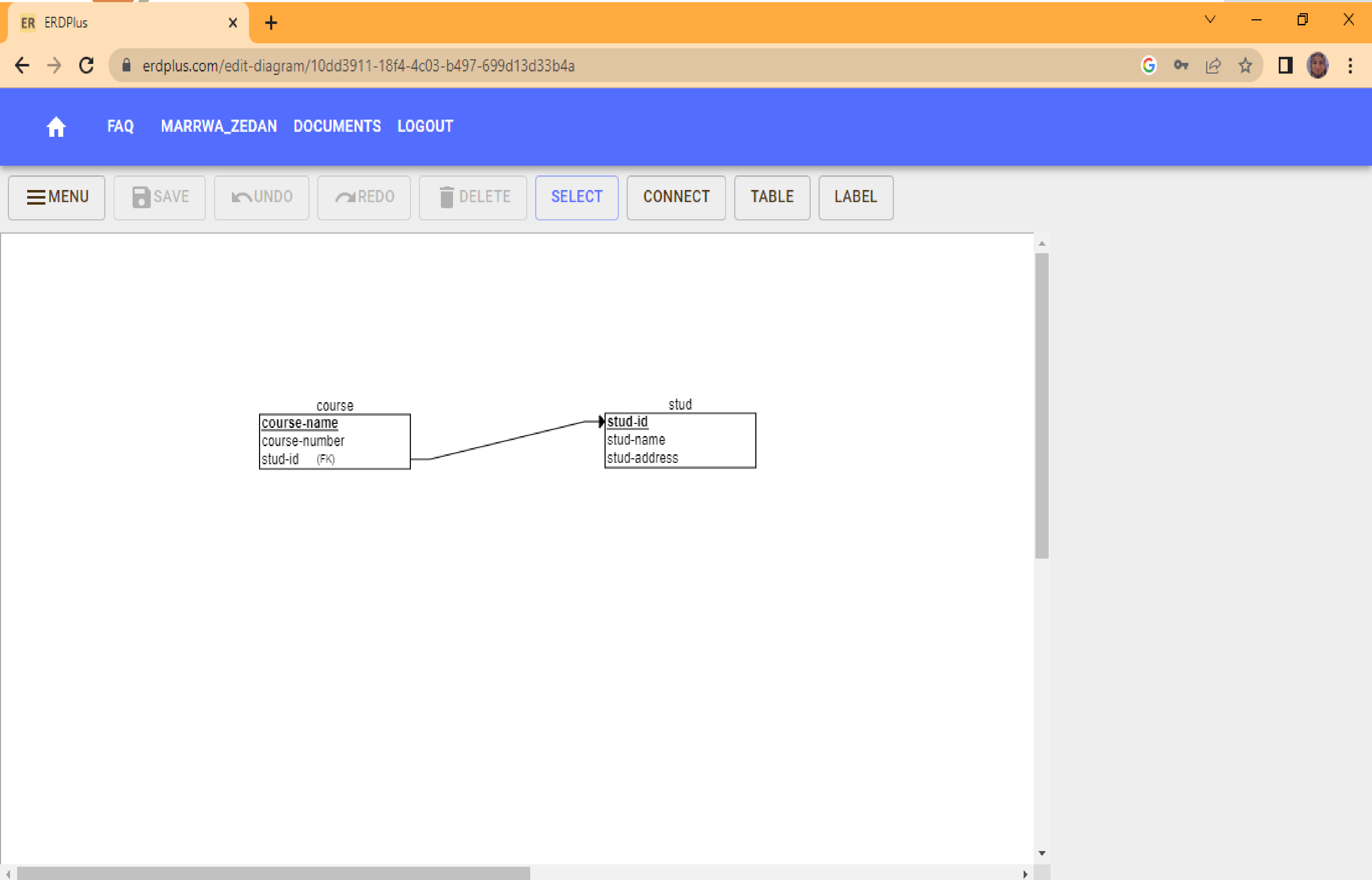
Each table in a relational schema is referred to as a **relation**. Rows in the table are called **tuples,** whereas the table columns are **attributes**. Tuples can be seen as the instances of an entity. A table can have many instances. For example, a school can have several students enrolled simultaneously. If student is an entity, then each student's record will be a tuple of that table (or relation).

ERD Plus



ERDPlus
erdplus.com/edit-diagram/7f207f97-b016-40c8-8706-c3bc7cb0cc38
FAQ
MARRWA_ZEDAN
DOCUMENTS
LOGOUT
MENU
SAVE
UNDO
REDO
DELETE
SELECT
CONNECT
TABLE
LABEL
book
book-id
book-name
Price
book_Authors
Authors
book-id (FK)
read
book-id (FK)
Stud-id (FK)
student
Stud-id
First name
Middle name
Last name
student_phone
phone
Stud-id (FK)

ERD Plus


ERDPlus
erdplus.com/edit-diagram/10dd3911-18f4-4c03-b497-699d13d33b4a
FAQ
MARRWA_ZEDAN
DOCUMENTS
LOGOUT
MENU
SAVE
UNDO
REDO
DELETE
SELECT
CONNECT
TABLE
LABEL
course
course-name
course-number
stud-id (FK)
stud
stud-id
stud-name
stud-address

THANKS