UNIVERSITY OF MINES AND TECHNOLOGY

Database Terminologies & Concepts

BY: DR ERIC AFFUM

Project

- You build a database application on your own.
- The domain of the application will be given or chosen.
- The application will have a simple but interactive interface.

Building an Application with a DBMS

Requirements modeling (conceptual, pictures)

- Decide what entities should be part of the application and how they should be linked.

Schema design and implementation

- Decide on a set of tables, attributes.
- Define the tables in the database system.
- Populate database

Write queries to the database (using SQL)

Presentation Outline

- ACID Properties
- Database Architecture
- Database Values
- Database Structures
- Database Relationships
- Database Constraints

ACID PROPERTIES

- A transaction is a group of database read and write operations that only succeeds if all the operations within succeed.
- Transactions can impact a single record or multiple records.

ACID PROPERTIES

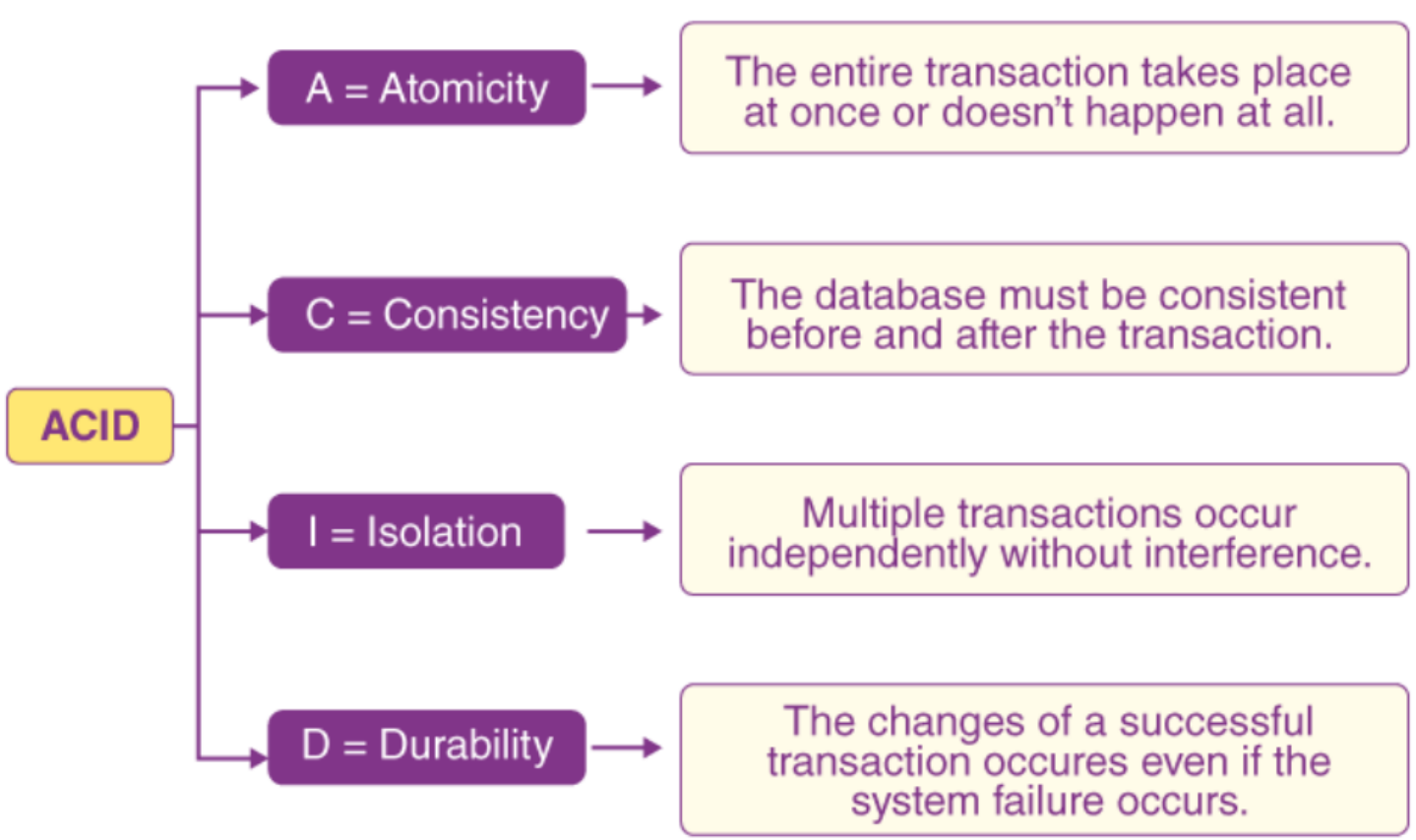
ACID is an acronym that refers to the set of 4 key properties that define a transaction:

- Atomicity
- Consistency
- Isolation
- Durability

If a database operation has these ACID properties, it can be called an ACID transaction.

Data storage systems that apply these operations are called transactional systems.

ACID PROPERTIES



ACID PROPERTIES

Atomicity

Atomicity means every transaction can be considered as a single unit, and they either run to completion or do not get executed at all.

The transactions can never occur partially/can't be midway.

Two operations here:

Commit: In case a transaction commits, the changes made are visible to us.

Abort: In case a transaction aborts, the changes made to the database are not visible to us.

ACID PROPERTIES

Before: A : 500	B : 200
Transaction T	
T1	T2
Read (A) A: = A - 100 Write (A)	Read (B) B: = B + 100 Write (B)
After: A : 400	B : 300

ACID PROPERTIES

Consistency

Consistency means that we have to maintain the integrity constraints so that any given database stays consistent both before and after a transaction.

Total after T occurs = 400 + 300 = 700.

Total before T occurs = 500 + 200 = 700.

ACID PROPERTIES

Isolation

Isolation ensures the occurrence of multiple transactions concurrently without a database state leading to a state of inconsistency.

A transaction occurs independently, i.e. without any interference.

Any changes that occur in any particular transaction would NOT be ever visible to the other transactions unless and until this particular change in this transaction has been committed or written to the memory.

ACID PROPERTIES

Durability

The durability property states that once the execution of a transaction is completed, the modifications and updates on the database gets written on and stored in the disk.

These persist even after the occurrence of a system failure. Such updates become permanent and get stored in non-volatile memory.

Thus, the effects of this transaction are never lost.

Database Architecture

The architecture for DBMSs is divided into three general levels:

- External
- Conceptual
- Internal

Database Architecture

External level of a database has the users' view of the database.

The conceptual level describes the logical structure of the entire database, including descriptions of data and relationships among the data.

The internal level gives the details of the physical storage of the database on the computer.

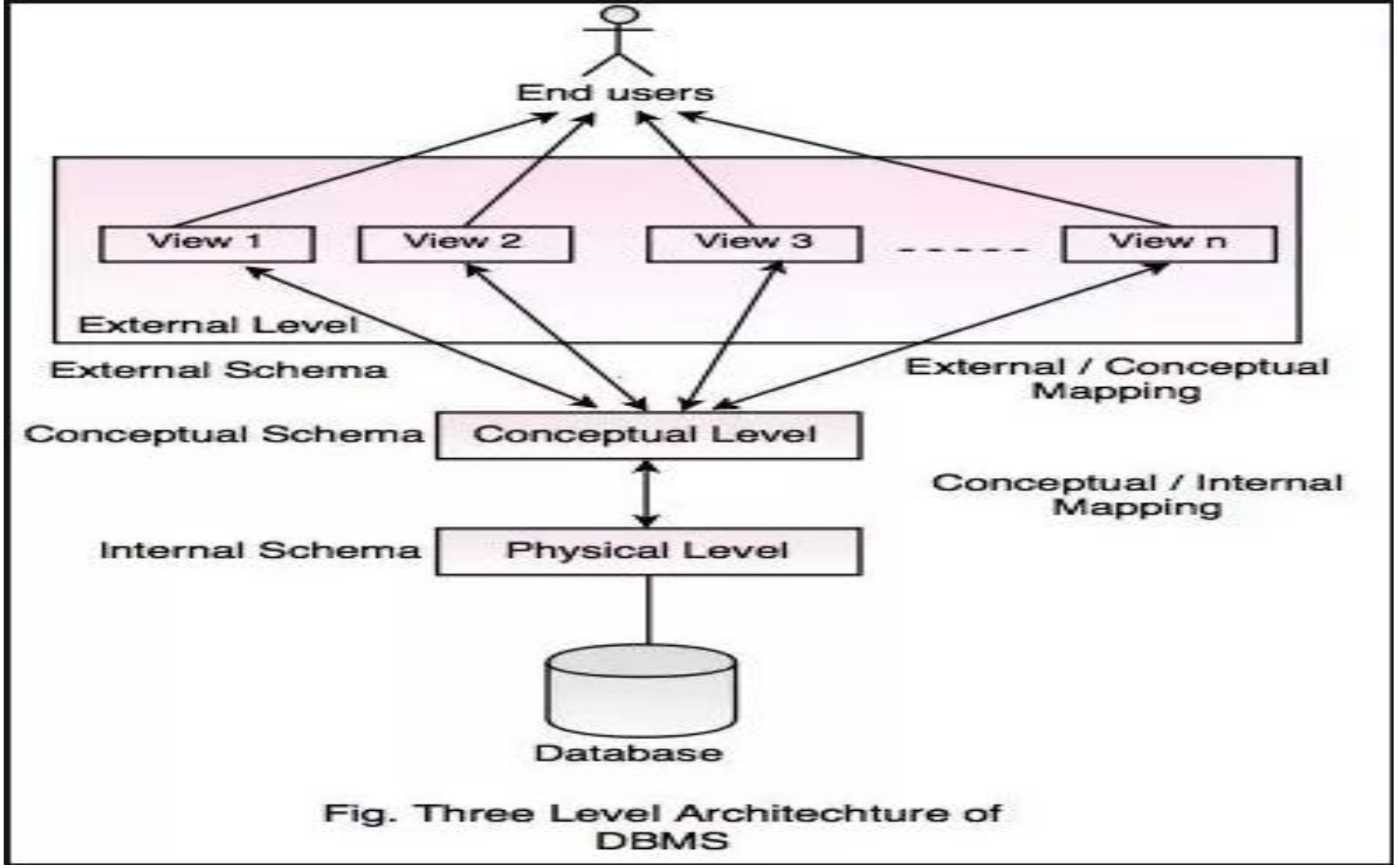


Fig. Three Level Architechture of DBMS

Properties of Relational Tables

- Values are Atomic
- Each Row is Unique
- Each Row is an instance of the Entity
- Column Values are of the Same Kind
- Each Column has a Unique Name
- Row and column order are not important

Terminologies used in RDBMS

A database can be modelled as:

- A collection of entities
- Relationship among entities
- And their attributes

RDBMS

Basically helps create and maintain relational databases.

There are 3 important elements in a RDBMS environment:

Field/Column/Attribute

Record/Row/Tuple

Keys(Primary key, Foreign key)

Terminologies used in RDBMS

There are four categories of terms used in Relational Databases and RDBMS:

- Value-related
- Structure-related
- Relationship-related
- Integrity-related

Value-Related Terms

- Data – Data is a raw fact
- Information – A data processed to give meaning
- Null- A null represents a missing or unknown value.

Structure-Related Terms

- Table – A table is a collection of data, arranged in rows (records) and columns (fields). A validation table (also known as a lookup table)- stores data that you specifically use to implement data integrity.

- Field- Known as an attribute in relational database theory is the smallest structure in the database and it represents a characteristic of the subject of the table to which it belongs.

Structure-Related Terms

- Record- Known as a tuple in relational database theory and represents a unique instance of the subject of a table.

- View- It is a "virtual" table composed of fields from one or more tables in the database; the tables that comprise the view are known as base tables.

Structure-Related Terms

- Keys (Primary, Foreign)

A primary key: It is a unique identifier of an instance within an entity. Examples: driver license number, telephone number, SSNIT number or vehicle identification number. A relational database must always have one and only one primary key

Structure-Related Terms

- Keys (Primary, Foreign)

Foreign key: is a column that is used to establish and enforce a link between the data in two tables. A table can contain a number of foreign keys.

One –to-One

- A one-to-one relationship between two entities indicates that each occurrence of one entity in the relationship is associated with a single occurrence in the related entity.
- There is a one-to-one mapping between the two, such that knowing the value of one entity gives you the value of the second.

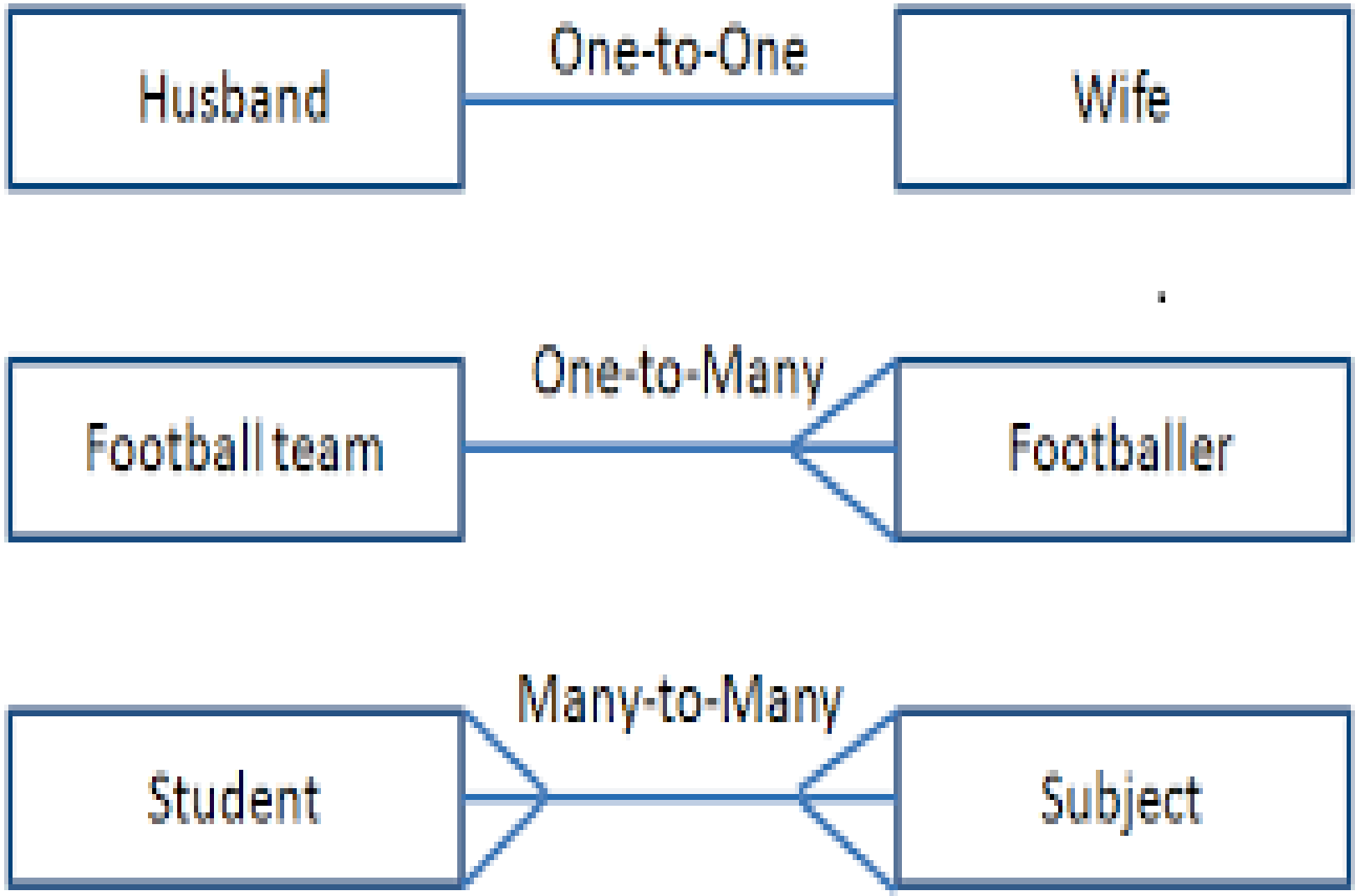
One-to-many

- A one-to-many or a many-to-one relationship between two entities indicates that a single occurrence of one entity is associated with one or more occurrences of the related entity.

Many-to-many

Many-to-many relationship between two entities indicates that either entity participating in the relationship may occur one or several times. This type of relationship is not practice in database. To resolve this a linking or join table is created

Relationship-Related Term



Data Integrity

Data integrity: refers to the validity, consistency, and accuracy of the data in a database.

- Table-Level integrity (entity integrity)
- Field-Level integrity (domain integrity)
- Relationship-Level integrity (referential integrity)
- Business rules (user-defined integrity)

Table Integrity

Table-Level integrity (entity integrity)

- Information contained in the table should pertain to the subject matter.
- Table must have a unique identifier that can be used to tell apart each record. This unique identifier is normally known as Primary Key of the table.

Field Integrity

Field/column-Level integrity (Domain integrity)

- Column integrity refers to the requirement that data stored in a column must adhere to the same format and definition. This includes data type, data length, default value of data, range of possible values, whether duplicate values are allowed, or whether null values are allowed.

Referential Integrity

Relationship-Level integrity (Referential integrity)

- Referential integrity is defined at the database design time and enforced by creating table relationships between tables. After the referential relationship is set up, database engine will follow the two rules stated above to guarantee data integrity. It will raise errors if the rules are violated.

Database Constraints

- A constraint in a database is a restriction placed at column or table level, a constraint ensures that your data meets certain data integrity rules.

- Database Constraints are declarative integrity rules of defining table structures.

Constraint Types

- **Data type constraint:** This defines the type of data, data length, and a few other attributes which are specifically associated with the type of data in a column.
- **Default constraint:** This defines what value the column should use when no value has been supplied explicitly when inserting a record in the table.
- **Nullability constraint:** This defines that if a column is NOT NULL or allow NULL values to be stored in it.

Constraint Types

- **Primary key constraint:** This is the unique identifier of the table. Each row must have a distinct value NULL values are not allowed in primary key values.
- **Unique constraint:** This defines that the values in a column must be unique and no duplicates should be stored.
- **Foreign key constraint:** This defines how referential integrity is enforced between two tables.

Constraint Types

- **Check constraint:** This defines a validation rule for the data values in a column so it is a user-defined data integrity constraint. This rule is defined by the user when designing the column in a table.

SUMMARY

Data integrity type	Enforced by database constraint
Row integrity	• Primary key constraint • Unique constraint
Column integrity	• Foreign key constraint • Check constraint • Default constraint • Data type constraint • Nullability constraint
Referential integrity	• Foreign key constraint
User-defined integrity	• Check constraint