



# RWANDA CODING ACADEMY

## *BASICS OF DATABASE DEVELOPMENT*

*by*

*Hilaire HATANGIMBABAZI*



# Why this course is selected?

Covers principles from other courses, such as

- ❑ Networking
- ❑ Website Design and Development
- ❑ Computer Programming



# Course Objectives

- ❑ Introduction to very basics
- ❑ Guides through different design stages
- ❑ Familiarize with tools
- ❑ Emphasis on design&development stages of database



# Lecture Objectives

- ❑ Some common uses of database systems.
- ❑ Characteristics of file-based systems.
- ❑ Problems with file-based approach.
- ❑ Meaning of the term database.
- ❑ Meaning of the term Database Management System (DBMS).



# Lecture Objectives

- ❑ Typical functions of a DBMS.
- ❑ Major components of the DBMS environment.
- ❑ Personnel involved in the DBMS environment.
- ❑ History of the development of DBMSs.
- ❑ Advantages and disadvantages of DBMSs.



# Examples of Database Applications

- ❑ Purchases from the supermarket
- ❑ Purchases using your credit card
- ❑ Booking a holiday at the travel agents
- ❑ Using the local library
- ❑ Taking out insurance
- ❑ Register at academic institution

# I. Introduction to Database

- **Def 1:** Database is an organized collection of logically related data
- **Def 2:** A database is a shared collection of logically related data that is stored to meet the requirements of different users of an organization
- **Def 3:** A database is a self-describing collection of integrated records

N.B: A database consists of both **data** and **metadata**.

# Introduction to Database (Cont)

- ❑ **Data:** stored representations of meaningful objects and events or
- ❑ Referred to facts concerning objects and events that could be recorded and stored on computer media
  - ❑ Structured: numbers, text, dates
  - ❑ Unstructured: images, video, documents

## **Examples:**

1. James
2. Rwanda
3. Instructor





## ***Introduction to Database(cont)***

- **Metadata:** Metadata is the data that describes the data's structure within a database.

The database stores **metadata** in an area called the **data dictionary**, which describes the **tables, columns, indexes, constraints**, and other items that make up the database.

- **Information:** data processed to increase knowledge in the person using the data

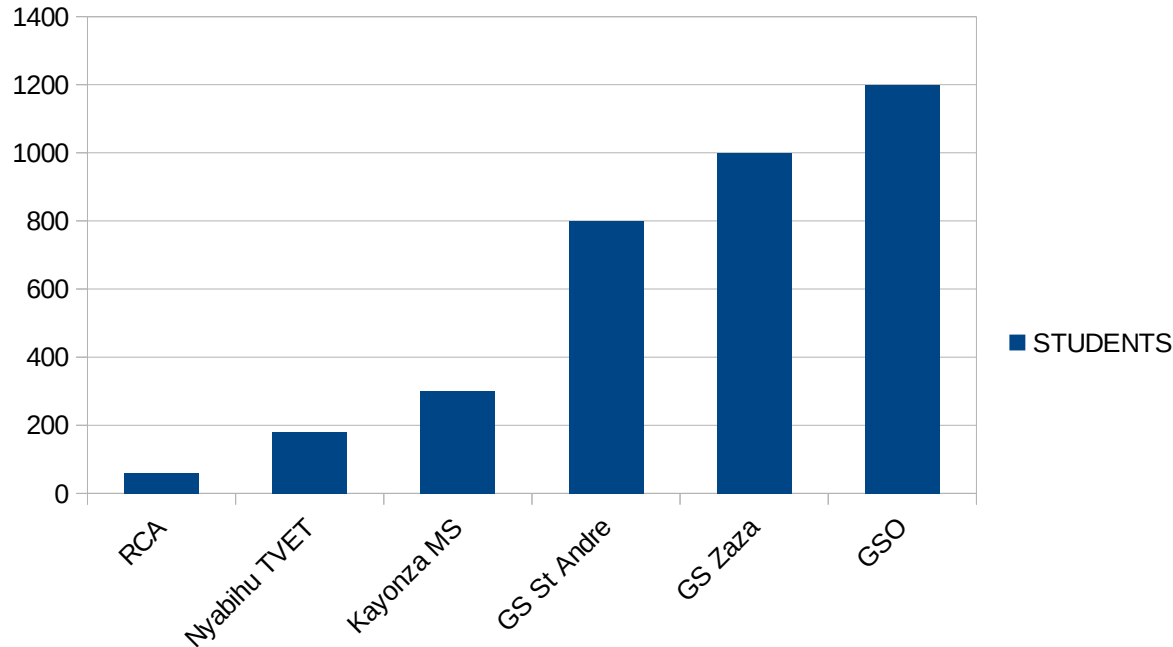
**Example1: James is Instructor from Rwanda**



## Example2: Data in Context

School	Students
RCA	60
Nyabihu TVET	180
Kayonza MS	300
GS St Andre	800

# Information in Context




Converting Data to Information. Graphical displays turn data into useful information that managers can use for decision making and interpretation

**Table 1-1** Example Metadata for Class Roster

<i>Data Item</i>			<i>Value</i>			
Name	Type	Length	Min	Max	Description	Source
Course	Alphanumeric	30			Course ID and name	Academic Unit
Section	Integer	1	1	9	Section number	Registrar
Semester	Alphanumeric	10			Semester and year	Registrar
Name	Alphanumeric	30			Student name	Student IS
ID	Integer	9			Student ID (SSN)	Student IS
Major	Alphanumeric	4			Student major	Student IS
GPA	Decimal	3	0.0	4.0	Student grade point average	Academic Unit

Descriptions of the properties or characteristics of the data, including data types, field sizes, allowable values, and data context



**II. Database Management System (DBMS):** A software package/ system to facilitate the creation and maintenance of a computerized database.

It

- defines (data types, structures, constraints)
  - construct (storing data on some storage medium controlled by DBMS)
  - manipulate (querying, update, report generation)
- databases for various applications.

**Database System:** The DBMS software together with the data itself. Sometimes, the applications are also included.



## II.1. Types of Database Management Systems

There are several types of database management systems. Here is a list of some common database management systems:



# Types of Database Management Systems (Cont)

1. Relational databases
2. Hierarchical databases
3. Network databases
4. Object-oriented databases
5. Graph databases
6. ER model databases
7. Document databases or NoSQL databases



# Types of Database Management Systems (Cont)

We can have a look on some types of DBMSs, but because in this course we will be limited to Relational Databases, we will use MySQL as the case study of Relational Database Management System (RDBMS)





# III. Relational databases

A relational database is a set of formally described **tables** from which data can be accessed or reassembled in many different ways without having to reorganize the database tables.



# Relational database(Cont)

Each table, which is sometimes called a ***relation***, in a relational database contains one or more data categories in ***columns***, or ***attributes***.



# Relational database(Cont)

Each row, also called a ***record*** or ***tuple***, contains a unique instance of data, or key, for the categories defined by the columns.  
(We'll discuss different types of keys later)



# Relational database(Cont)

Each table has a unique primary key, which identifies the information in a table. The relationship between tables can then be set via the use of foreign keys -- a field in a table that links to the primary key of another table.



# Table

“A *table* is the primary unit of physical storage for data in a database.”<sup>1</sup>

Usually a database contains more than one table.

# Table

<b>Name</b>	<b>Company</b>	<b>Phone Number</b>	<b>E-mail Address</b>
Vedat Diker	CLIS/UMD	(301) 405 9814	vedat@umd.edu
Bugs Bunny	Acme, Inc.	(123) 555 9876	bugs@acme.com
Will E. Coyote	Acme, Inc.	(123) 555 9821	will@acme.com

# Single table Database

ISBN	Title	AuID	AuName	AuTel	PubID	PubName	PubTel	Price
0-99-999999-9	Emma	1	Austen	111-111-1111	1	Big House	123-456-7890	20.00 zł
0-91-335678-7	Faerie Queen	7	Spenser	777-777-7777	1	Big House	123-456-7890	17.00 zł
0-91-045678-5	Hamlet	5	Shakespeare	555-555-5555	2	Alpha Press	999-999-9999	20.00 zł
0-103-45678-9	Iliad	3	Homer	333-333-3333	1	Big House	123-456-7890	25.00 zł
0-555-55555-9	Macbeth	5	Shakespeare	555-555-5555	2	Alpha Press	999-999-9999	12.00 zł
0-55-123456-9	Main Street	10	Jones	123-333-3333	3	Small House	714-000-0000	23.00 zł
0-55-123456-9	Main Street	9	Smith	123-222-2222	3	Small House	714-000-0000	23.00 zł
0-12-333433-3	On Liberty	8	Mill	888-888-8888	1	Big House	123-456-7890	25.00 zł
0-321-32132-1	Balloon	2	Sleepy	222-222-2222	3	Small House	714-000-0000	34.00 zł
0-321-32132-1	Balloon	4	Snoopy	444-444-4444	3	Small House	714-000-0000	34.00 zł
0-321-32132-1	Balloon	11	Grumpy	321-321-0000	3	Small House	714-000-0000	34.00 zł



# Disadvantages of a single table database

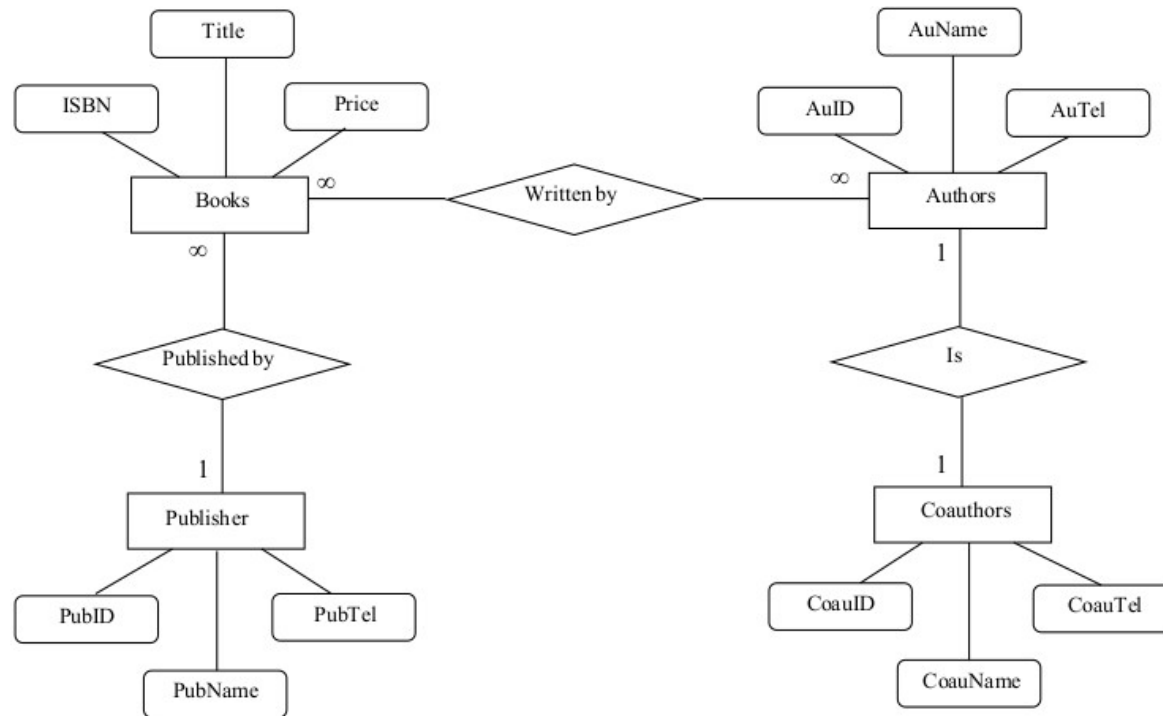
- Redundancy of data
- Problem with complex data
- Problems in updating in bulk (new phone number)
- Problems in adding incomplete data (new publisher)
- Problems in removing group of data (all books from the publisher)

**Solution:**

Relational Database Management System (RDBMS)



# Example of relations in Database



# Library Database-Books table

ISBN	Title	PubID	Price
0-103-45678-9	Iliad	1	25.00
0-11-345678-9	Moby Dick	3	49.00
0-12-333433-3	On Liberty	1	25.00
0-123-45678-0	Ulysses	2	34.00
0-12-345678-9	Jane Eyre	3	49.00
0-321-32132-1	Balloon	3	34.00
0-55-123456-9	Main Street	3	23.00
0-555-55555-9	Macbeth	2	12.00
0-91-045678-5	Hamlet	2	20.00
0-91-335678-7	Faerie Queen	1	15.00
0-99-777777-7	King Lear	2	49.00
0-99-999999-9	Emma	1	20.00
1-1111-1111-1	C++	1	30.00
1-22-233700-0	Visual Basic	1	25.00

# Library Database-Authors table

AuID	AuName	AuTel
1	Austen	111-111-1111
2	Melville	222-222-2222
3	Homer	333-333-3333
4	Roman	444-444-4444
5	Shakespeare	555-555-5555
6	Joyce	666-666-6666
7	Spenser	777-777-7777
8	Mill	888-888-8888
9	Smith	123-222-2222
10	Jones	123-333-3333
11	Snoopy	321-321-2222
12	Grumpy	321-321-0000
13	Sleepy	321-321-1111



# Library Database-Publishers table

PubID	PubName	PubTel
1	Big House	123-456-7890
2	Alpha Press	999-999-9999
3	Small House	714-000-0000

# Library Database-Books/Author table

ISBN	AutID
0-103-45678-9	3
0-11-345678-9	2
0-12-333433-3	8
0-123-45678-0	6
0-12-345678-9	1
0-321-32132-1	11
0-321-32132-1	12
0-321-32132-1	13
0-55-123456-9	9
0-55-123456-9	10
0-555-55555-9	5
0-91-045678-5	5
0-91-335678-7	7
0-99-777777-7	5
0-99-999999-9	1
1-1111-1111-1	4
1-22-233700-0	4



# IV. Database Keys

Keys are very important part of Relational database model. They are used to establish and identify relationships between tables and also to uniquely identify any record or row of data inside a table.



# Database Keys (Cont)

A Key can be a single attribute or a group of attributes, where the combination may act as a key.



# Why we need a Key?

Keys are defined to easily identify any row of data in a table.

Let's try to understand about all the keys using a simple example.



# Why we need a Key?(Cont)

student_id	name	phone	age
1	Akon	9876723452	17
2	Akon	9991165674	19
3	Bkon	7898756543	18
4	Ckon	8987867898	19
5	Dkon	9990080080	17

Let's take a simple **Student** table, with fields `student_id`, `name`, `phone` and `age`.

## IV.1. Super key

Super Key is defined as a set of attributes within a table that can uniquely identify each record within a table. Super Key is a super set of ***Candidate key***.

In the table defined above super key would include ***{student\_id}, {student\_id, name}, {phone}, {student\_id, name, phone}, , {student\_id, name, phone, age}***.



## IV.2.Candidate Key

Candidate key is a minimal super key with no redundant attributes.

There can be more than one candidate key.  
In our example, ***student\_id*** and ***phone*** both are candidate keys for table Student.



## IV.3.Primary Key

Primary key is a candidate key that is most appropriate to become the main key for any table. It is a key that can uniquely identify each record in a table like any other super or candidate key.



## IV.4.Composite Key

Key that consists of two or more attributes that uniquely identify any record in a table is called ***Composite key***. But the attributes which together form the ***Composite key*** are not a key independently or individually.

# Composite key (Cont)

Composite Key



student_id	subject_id	marks	exam_name



## IV.5.Secondary or Alternative key

The candidate key which are not selected as primary key are known as ***secondary keys*** or ***alternative keys***.



## IV.6.Non-key Attributes

**Non-key** attributes are the attributes or fields of a table, other than super or candidate key attributes/fields in a table.

Ex: ***{name}, {age}***





# Prime Attributes

Attributes that form a candidate key of a relation, i.e. attributes of candidate key, are called prime attributes.



## IV.7.Foreign key

Foreign keys are the columns of a table that points to the primary key of another table. They act as a cross-reference between tables, and they are used to make one of the following relationships between two tables:

# Foreign key example

In the below example the Stu\_Id column in Course\_enrollment table is a foreign key as it points to the primary key of the Student table.

Course_Id	Stu_Id
C01	101
C02	102
C03	101
C05	102
C06	103
C07	102



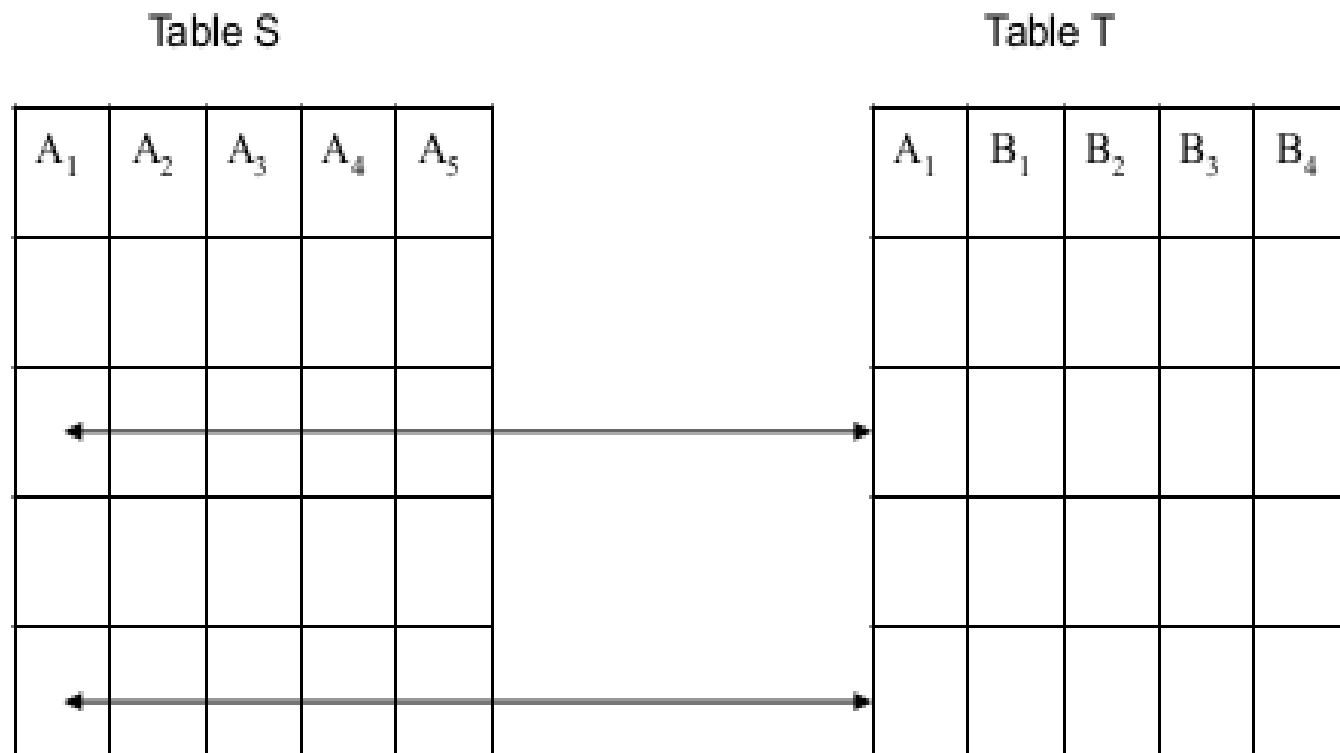
# **V. Types Relationship between tables**

One-to-one

One-to-many

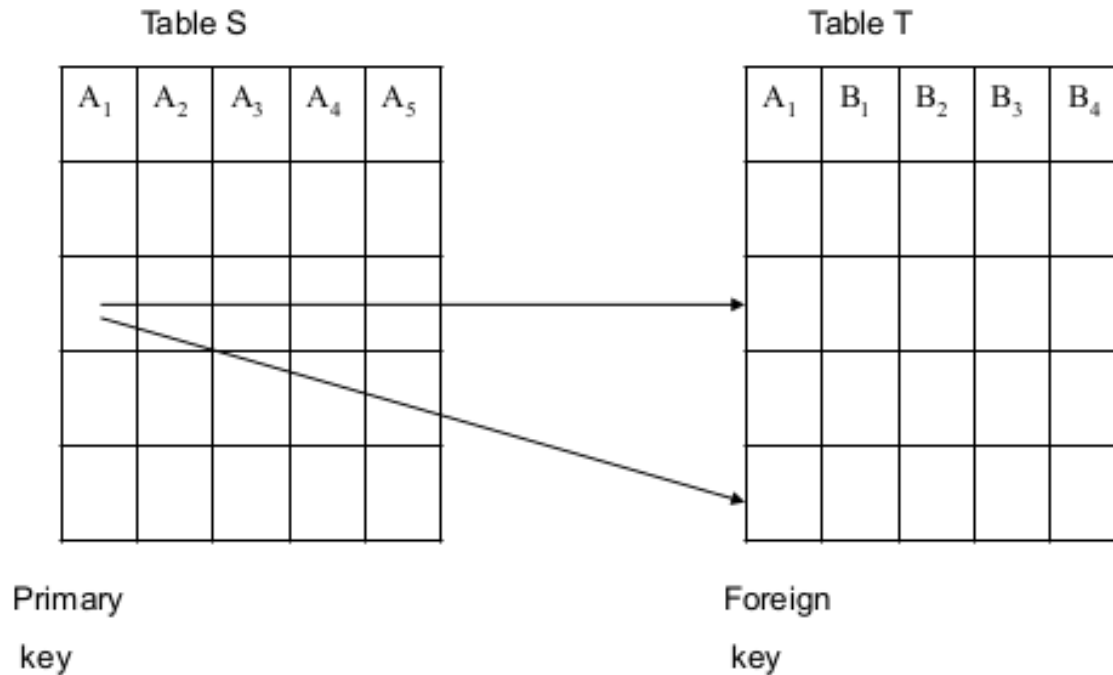
Many-to-many

# V.1. One to one Relation



In a one-to-one relationship, one record in a table is associated with one and only one record in another table.

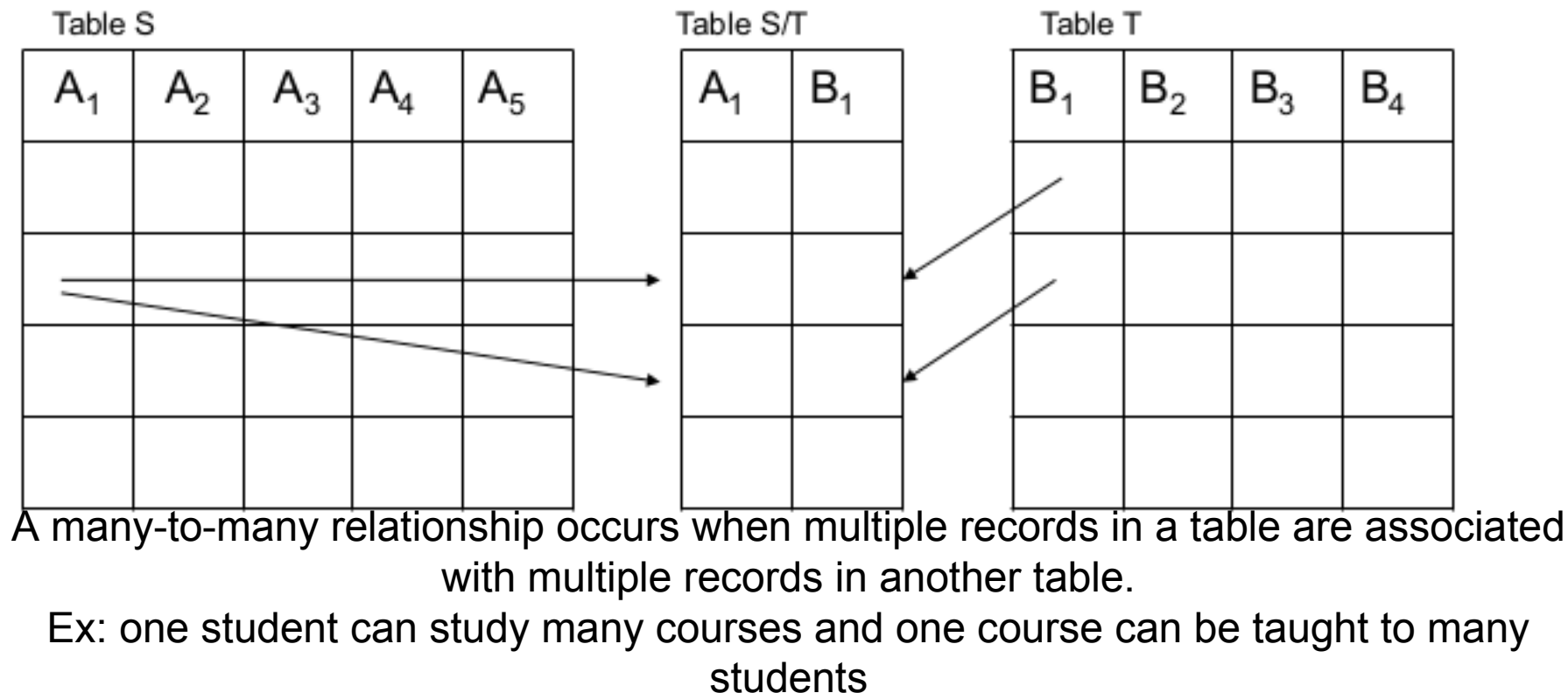
## V.2. One to many relation



Values of the foreign key can not be different from the values of the primary key.

In a one-to-many relationship, one record in a table can be associated with one or more records in another table

## V.3. Many to many Relation





# Enforcing referential integrity

- Cascade Update Related Fields – the values of foreign keys change following changes of the values of the primary key
- Cascade Delete Related Records – deleting a record from the primary field in a relationship causes a deletion of all related records in the second table





# **VI. Data Types for attributes**

When creating a table, you need to specify the data type for each column. Some commonly used data types are listed below.



# Data Types for attributes (Cont)

1. **Character** data types: Character data consist of any combination of letters, symbols, and numeric characters.
  - a. **CHAR**: It is a fixed-length character data type.  
Example: To specify the column, Street, and to have the CHAR data type with length 30, use **Street CHAR(30)**



# Data Types for attributes (Cont)

**b. VARCHAR:** It is a variable-length character data type.

Example: To specify the column, Description, and to have the VARCHAR data type with the maximum length of 300, use  
Description VARCHAR(300)



# Data Types for attributes (Cont)

2. Number data types: Number data types include **integers** and **decimals**.
  - a. INT: The INT data type consists of positive, zero, and negative whole numbers. The range of INT is from  $-2,147,483,648$  to  $2,147,483,647$ .

Example: Integers can be used as the data type for a key column.



# Data Types for attributes (Cont)

b. DECIMAL (or NUMERIC): The DECIMAL data type includes positive and negative decimal numbers. The values of DECIMAL data are from  $-10^{38} + 1$  through  $10^{38} - 1$ .



# Data Types for attributes (Cont)

3. Date and Time data type: It represents date and time values.

a. **DATETIME**: This data type specifies a column to contain date and time values from

1-1-1753 to 12-31-9999



# Data Types for attributes (Cont)

4. Monetary data type: It represents amounts of money.

a. MONEY: This data type specifies a column to contain money values.



## VII. Database Constraints

- Database constraints are restrictions on the contents of the database or on database operations
- A constraint is a rule enforced by the database manager to limit the values that can be inserted, deleted, or updated in a table.
- Database constraints provide a way to guarantee that:
  - rows in a table have valid primary or unique key values
  - rows in a dependent table have valid foreign key values that reference rows in a parent table
  - individual column values are valid



# Database Constraints (Cont)

MySQL DBMS supports the following constraints:

- **primary key constraint** (to enforce existence integrity)
- **foreign key constraint** (to enforce foreign key, or referential integrity)
- **unique constraint** (to enforce candidate key integrity)
- **check constraint** (to restrict a column's values; a partial enforcement of domain integrity)
- **Not null**
- **Default**
- You specify one or more of these constraints when you use the **Create Table** statement to create a base table.
- You can also **add** or **drop** constraints with the **Alter Table** statement.

# Database Constraints (Cont)

CONSTRAINT	DESCRIPTION
NOT NULL	In MySQL NOT NULL constraint allows to specify that a column can not contain any NULL value. MySQL NOT NULL can be used to CREATE and ALTER a table.
UNIQUE	The UNIQUE constraint in MySQL does not allow to insert a duplicate value in a column. The UNIQUE constraint maintains the uniqueness of a column in a table. More than one UNIQUE column can be used in a table.
PRIMARY KEY	A PRIMARY KEY constraint for a table enforces the table to accept unique data for a specific column and this constraint creates a unique index for accessing the table faster.
FOREIGN KEY	A FOREIGN KEY in MySQL creates a link between two tables by one specific column of both tables. The specified column in one table must be a PRIMARY KEY and referred by the column of another table known as FOREIGN KEY.
CHECK	A CHECK constraint controls the values in the associated column. The CHECK constraint determines whether the value is valid or not from a logical expression.
DEFAULT	In a MySQL table, each column must contain a value ( including a NULL). While inserting data into a table, if no value is supplied to a column, then the column gets the value set as DEFAULT.



## VII.1. Primary Key Constraints

A primary key serves as the unique identifier for rows in the table.

Each primary key column's definition must include **Not Null**.

For a table with a primary key constraint, DBMS blocks any attempt to insert or update a row that would cause two rows in the same table to have identical value(s) for their primary key column(s).

A table definition can have no more than one primary key constraint.



## VII.2. Unique Constraints

A **unique constraint** is similar to a primary key constraint

It doesn't have to be defined with Not Null.

## VII.3. Foreign Key Constraints

- A **foreign key constraint** specifies how records in different tables are related and how a DBMS should handle row insert, delete, and update operations that might violate the relationship.
- For example, sales rows are generally related to the customers who place the orders. Although it might be valid for a customer row to exist without any corresponding sale rows, it would normally be invalid for a sale row not to have a reference to a valid customer.
- With a relational DBMS, the relationship between rows in two tables is expressed by a **foreign key** in the **dependent table**. A foreign key is one or more columns that contain a value identical to a primary key (or unique key) value in some row in the **parent table** (i.e., the **referenced table**).



## VII.4. Check Constraints

Check constraints are used *to enforce the validity of column values*.

For example, a check constraint can compare a column to a constant to another column in the same table or to an expression.

DBMSs check to make sure a new or changed row doesn't violate any of its table's check constraints before an insert or update operation is allowed.



## VII.5. Not Null constraint

A column defined with a NOT NULL constraint requires that for every row entered into the table, there must be a value for that column.

For example, if the column in an employees table was defined as NOT NULL, every employee entered into the table **MUST** have a value in the column.



## VII.6. Default constraint

The DEFAULT constraint is used to insert a default value into a column.

The default value will be added to all new records, if no other value is specified.

Defaults can't be used on columns that are automatically increasing or changing like timestamp and identity columns. However, they can be used on duplicating rows.





# Practical Exercise:

Make group of four students and each group design any database in the real life. On designed database apply necessary DB constraints have to be applied. Those constraints must be shown in data dictionary of designed database or in its tables' schemes.



# References

1. Robert J, Robbins, Database Fundamentals
1. Celko, J. (2005). SQL Programming Style. San Francisco, CA: Morgan Kaufman Publishers.
2. Date, C. J., & McGoveran, D. (1995). The principle of Orthogonal Design. In C. J. Date, & D. McGoveran, Relational Database Writrings 1991-1994. Addison Wesley.
3. Nadkarni, P. M. (2002). An Introduction to EAV systems. National GCRC Meeting. Baltimore, MD.
4. Pascal, F. (2005, March). What First Normal Form really means and means not. Retrieved Oct 1, 2009, from Database Debunkings:  
<http://www.dbdebunk.com/page/page/629796.htm>
5. Peterson, D. (2006, March 24). Lookup Table Madness. Retrieved September 26, 2009, from SQL Server Central:  
<http://www.sqlservercentral.com/articles/Advanced/lookuptablemadness/1464/>
6. Teorey, T. J. (1994). Database Modeling & Design: The Fundamental Principles. Morgan Kaufmann.