SUCCEED WE MUST

DATABASE PROGRAMING BIT2212

Lecture 1 – Monday 5th February 2017

Mwavu Rogers
Email: rmwavu@must.ac.ug
Phone 0773426328 and 0700497421
Faculty of computing and informatics
DEPARTMENT OF INFORMATION TECHNOLOGY

SUCCEED WE MUST

Lecturing Venues and Course Delivery methods

- Lecturing Venue: Lab 2 Tuesday 3:00pm-05:00am and lab2 Wednesday 2:00pm-4:00pm
- **Course Delivery methods**
- Lectures
- Individual or group research projects
- Hands-on assignments
- Class Presentations
- Tutorials

Course Evaluation

- **Evaluation**
  - Coursework (40%)
    - Assignments and class participation (15%)
    - Practical Tests (20%)
    - Class Attendance (5%)
  - Exam (60%)

Introduction – Course Objectives

By the end of this course, students should have knowledge on how;

- To design well structured Logical Database designs (Requirements gathering, Normalization, EER diagrams)
- To implement the Logical Database Design to a physical database structure using the De-factor SQL standard
- To develop Relational databases using MySQL, Oracle and Access
- To have knowledge on Database Connectivity using various programming languages and drivers (Php and Mysql, ODBC,JDBC)
- To introduce students to potential research areas in the field.
- To highlight to students the various career openings, the knowledge of Databases can provide.

SUCCEED WE MUST

Reference Materials and Books

- Ramez. E, Shamkant B. N, *Database System: Models, Languages, design and Application Programming.* 6th Edition
- Peter .R, Carlos C, Keeley .C, *Database Systems: Design, Implementation and Management*
- Abraham. S, Henry F. K, Sudarshan S, *Database System Concepts,* 5th Edition
- Micheal .M, *Oracle Database: 11g – PL/SQL Programming*
- Micheal. V, *Database design, Application, Development and Administration,* 3rd Edition
- http://lms.must.ac.ug

Major DBMS today

- Oracle
- MySQL (acquired by Sun, then Oracle)
- IBM DB2 (from System R, System R*, Starburst)
- Microsoft SQL Server (MSSQL)
- Teradata
- Sybase (acquired by SAP)
- Informix (acquired by IBM)
- PostgreSQL (from UC Berkeley's Ingres, Postgres)
- Tandem NonStop (acquired by Compaq, now HP)
- SQLite
- Microsoft Access
- BerkeleyDB (acquired by Oracle)

Today's Content

The Logical Design of a Database is the foundation block for the effective implementation of a good Database.

Note: To design a good logical design you don't need to go anywhere near a Computer! While designing the logical design of the database, We do not consider the of relational database management system

Today's Content continued

To achieve our goal we shall cover the following:

- The Logical Structure of a Database
  - Understanding Requirements
  - Table Design & Normalisation
  - Enhanced Entity Relationship Diagrams

The Logical Structure

A Logical Database structure is the design of a Database based on the requirements stipulated by the Client.

The **logical database design** phase formalizes the objects, or entities, and their relationships.

Another primary task of logical **design** is to ensure that the modeled entities are modified by attributes that uniquely pertain to them.

Cont……

The requirements analysis phase of the design process reveals the real world objects and their attributes that the database must represent as well as the relationships among them.

This involves the function, complexity, scope and business practices which relate to the Project.

By developing a Logical Model all parties can ensure that each and every requirement is fulfilled and that no requirement is misunderstood.

How is a Logical Structure Developed : Stage 1

Business practices / processes are mapped by

- Assessing and recording existing Manual Processes, through:
  - Floor Walking,
  - Talking to ALL interested parties (Interviewing)
  - Carrying out test scenarios based on the Businesses operations.
  - Collecting existing documentation, forms and written processes
  - Observations
  - Journey Mapping(read more about walker's approach)
  - Brainstorming

How is a Logical Structure Developed: Stage 1

Business practices / processes are mapped by

- Process can be illustrated by use of .
  - »Unified Modelling Language (UML)
  - »Flowcharts
  - »Block diagrams
- All these can be added in the requirements document

SUCCEED WE MUST

How is a Logical Structure Developed: Stage 1

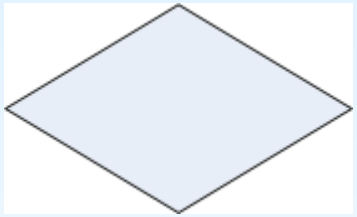
Traditional Flowcharts are a very simplistic way of illustrating processes as they typically involve 3 symbols (although there are many more symbols available).

The 3 key symbols are:

1. Terminators (Start / End)

2. Processes

3. Decisions



How is a Logical Structure Developed: Stage 1

UML is more complex, but is supposed to be used throughout an entire project lifecycle, and has different “layers” / complexities which allow each individual within the project to understand content in relation to their relevant context.

UML is also closely related to Object Oriented methodologies.

Data base design Vernacular

Logical design	Physical Design	Theoretical
Entity	Table	Relational
Attribute	Colum, field ,	Attribute
	row , records	Tuple

How is a Logical Structure Developed: Stage 2

Once processes have been recorded and agreed, it is essential to define the attributes and interrelationships between them

For example, using the Scenario of a Patient we need to collect many different attributes, such as

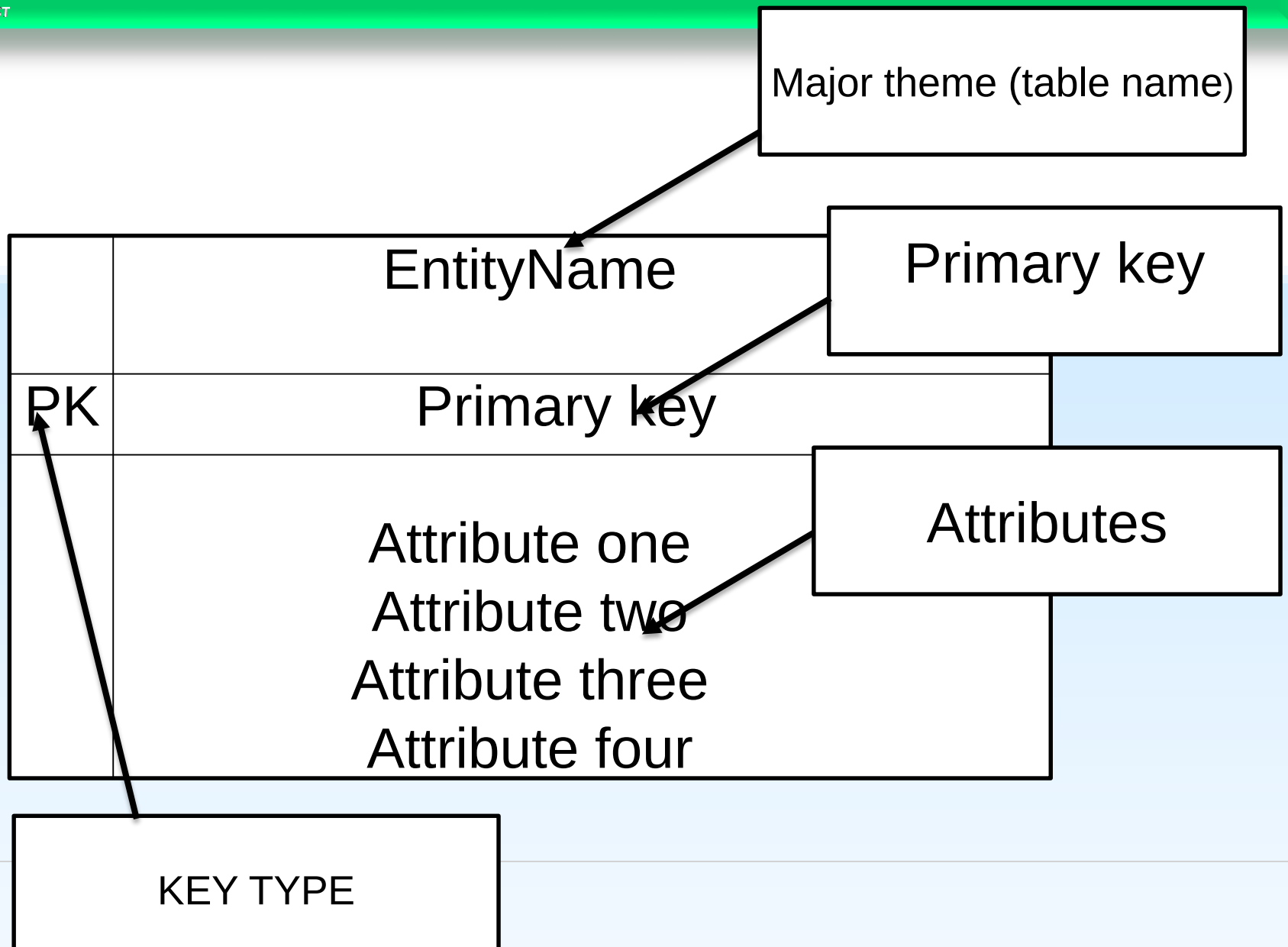
- Demographic Information (Name, Age, Sex, Date of Birth, Birthplace, Nationality, Ethnic Origin),
- Medical Information (Registered Doctor, Known Problems, Medications) and so on

SUCCEED WE MUST

ENTITY RELATIONSHIP DIAGRAM

- ERD- Standard method for drawing entities, relationships and attributes
- Process
- Analysis should reveal nouns(person , place or a thing)
- Nouns are group to a common theme(entities)
- Other nouns describe common themes (Attributes)
- Some nouns uniquely identify a specific instance of an entity (primary key)

ENTITY


Major theme (table name)
EntityName
Primary key
PK
Primary key
Attributes
Attribute one
Attribute two
Attribute three
Attribute four
KEY TYPE

Entity Types

Entity Roles	Description
Domain	Entity describing a core business element of the database
Linking	Entity used to resolve a many-to-many relationship into two one-to-many relationships
Lookup	Entity used to store lookup values and help ensure data integrity and consistency
Weak	An entity that depends on another entity for its meaning

How is a Logical Structure Developed: Stage 2

We can represent this in a table:

Name	Sex	DOB	Phone	Mobile	Birthplace	Country of Origin	Doctor
Opio	Male	02/03/1980	041 495948	0781 23949	Mbarara	Uganda	Dr Mario Balotery
Mary	Female	05/06/1986	042 49583		Kampala	UK	Dr Allen
Tracy	Female	03/07/1987		0789 43993	Gulu	Uganda	Dr Song Belong
Greg	Male	04/06/1975	040 49403		Mbarara	Uganda	Dr Peter
Owen	Male	02/02/1954	041 4930	0765 49322	Gulu	Uganda	Dr Smith
Tina	Female	06/07/1993		0765 24590	Liverpool	UK	Dr Smith

How is a Logical Structure Developed: Stage 2

In a manual process this is acceptable but in a database we want to follow these rules:

- Data redundancy should be minimised or removed
- Each record should have a unique identifier, which makes it easier and faster to access
- Data in a table should only store information for a single type of entity, i.e. There should be separate tables Patients & Professional Carers (Doctors, Nurses etc)
- As few nullable columns as possible should exist (columns that contain nothing). n.b. Blank columns and NULL columns are not necessarily the same
- There should be no repeating rows or columns. e.g There should not be several columns for different phone numbers

How is a Logical Structure Developed: Stage 2

This is called Normalisation, for example:

Patients						
Patient_ID	**Name**	**Sex**	**DOB**	**City_ID (Birthplace)**	**Country_ID (Country of origin)**	**Carer_ID (Doctor)**
1	Opio	Male	02/03/1980	1	1	1
2	Mary	Female	05/06/1986	2	2	2
3	Tracy	Female	03/07/1987	3	1	1
4	Greg	Male	04/06/1975	1	1	2
5	Owen	Male	02/02/1954	3	1	3
6	Tina	Female	06/07/1993	4	2	3

How is a Logical Structure Developed: Stage 2

There are now additional tables to reduce repetition

Cities		
City_ID	**City**	**Country_ID**
1	Mbarara	1
2	Kampala	1
3	Gulu	1
4	Liverpool	2

Countries	
Country _ID	**City**
1	Uganda
2	UK
3	Spain
4	Kenya

Personal_Carers	
Carer _ID	**Carer_Name**
1	Dr Mario Balotery
2	Dr Peter
3	Dr Song belong
4	Dr Baryashaba

How is a Logical Structure Developed: Stage 2

and a **Phone_Numbers** table to eradicate repeating Columns and an additional table for the **Number_Type**

Phone_Numbers			
Phone_ID	**Phone_Number**	**Patient_ID**	**Type_ID**
1	041 495948	1	1
2	0781 23949	1	2
3	042 49583	2	1
4	0789 43993	3	2
Etc…			

Number_Type	
Type_ID	**Type**
1	Phone
2	Mobile
3	Fax

SUCCEED WE MUST

How is a Logical Structure Developed: Stage 2

The following attributes were taken into account when defining the previous content:

- Tables - A database may contain only one table or as many tables as you like. You refer to a table by its name. A table contains information or records about something in the real world (an entity) such as patients, referrals, consultants.
- Rows - A table can contain many rows of information. One row could hold information about ONE particular patient or consultant.
- Columns – A table is made up of vertical columns of data. You refer to a column by its name. A column has a specific data type and size. It contains one important piece of information , for example the SURNAME of an patient.

How is a Logical Structure Developed: Stage 2

The following attributes were taken into account when defining the content:

- Primary Keys – A primary key is a column that defines the uniqueness of a row, for example the Patient ID. You can create one Primary Key per table so that there can be no other rows with the same Patient ID. Primary keys also enforce database relational integrity rules.
- Foreign Keys – A Foreign Key is a column that defines how tables relate to each other, for example the Patient ID in the Patients table could link to another table (the Addresses Table).
- Fields – At the intersection of a row and column is a field. A field may or may not contain data. If it does not contain data it is said to contain a NULL value. A field holds one item of information for one row in a table.

How is a Logical Structure Developed: Stage 2

Once a table has been defined the following should be true:

- There should be no duplicate rows. Each row is uniquely identified by a Primary Key.
- There must be no duplicate column names
- Row order is insignificant . The default order is the sequence in which rows are inserted
- Column order is insignificant when storing data as the order is specified in the query. ie. When data is retrieved.
- Field values are atomic. That is they cannot be broken down into smaller components, are non-decomposable

How is a Logical Structure Developed: Stage 2

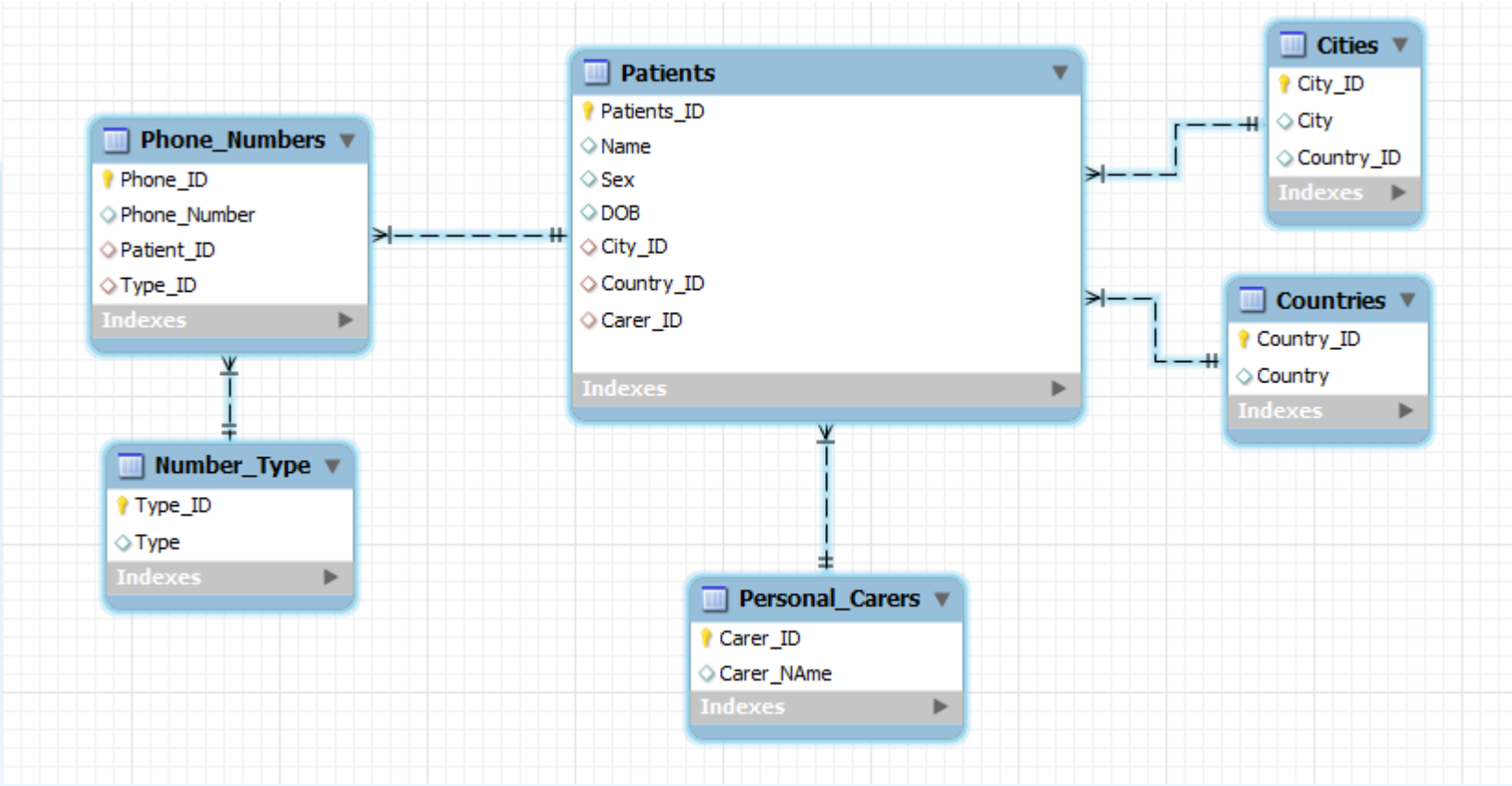
Finally, it is recommended that the following rules are followed to ensure a Database is easy to understand

- The name of a field should be Logical
- A Primary Key / Foreign Keys name should relate to the name of the table. i.e. If a table is called Patients its Primary Key should be called Patients_ID. Where record related to a Patient appears in another table the Foreign Key should also be called Patients_ID.
- Spaces should NOT be used when naming field, an underscore should be utilised. SQL can be used to hide the underscore from the User if required.

SUCCEED WE MUST

How is a Logical Structure Developed: Stage 2

Together the tables look like this:



How is a Logical Structure Developed: Stage 2

The links between the tables are joins, via Primary Key, Foreign Key relationships, such links (joins) can be:

- One to One – Where a record from table a directly relates to one record in table b
- One to Many – Where a record from table a directly relates to many records in table b
- Many to Many – Where many records from a table relates to many records in table b

How is a Logical Structure Developed: Stage 2

The links between the tables are joins, via Primary Key, Foreign Key relationships, such links (joins) can be:

- One to One – Where a record from table a directly relates to one record in table b
- One to Many – Where a record from table a directly relates to many records in table b
- Many to Many – Where many records from a table relates to many records in table b

SUCCEED WE MUST

Brains at work

Based on what we have learnt today, As a class of Database programming let us try to come up with a database for our class.

SUCCEED WE MUST

Exercise (Do not hand in)

- Come up with a group of five students and do this assignment. you are required to gather requirements for A students Academic Records Management System. document them including the various stakeholders and produce a **requirements document, flowchart and a logical(Normalised) design or EER diagram of your database**.
  - **You can have a minimum of 15 tables in your EER diagram**
-