Database Systems
Lab 1
Beginning with Database
Version 1.0

In this particular lab session, as you are very new to the database arena, the very basics and fundamentals of database will be revealed towards you. With this session, you will be able to know various things- what should be modelled with database, how can be a database created, how will you harmonize the real world things with database, etc.

At first, think- what will you try to put in database. Many things! Say, you are trying to keep records of a university's teachers, students and staffs. What will you do? In the very beginning you can at least think-

- The database will be a university database,
- The tables inside that university database will be teachers, students and staffs.
- Then after this primary thinking, you will advance yourself- what records will be kept.

The name of the records will be the columns of those tables. The entries into those tables will be the rows of those tables. These are easy tasks so far.

So, let us try to simulate that database- of course hypothetically! No computer based practical for the first day (hurrah! ☺).

1. We know we are focusing on keeping records of personal information of a university's teachers, students and staffs. Then the database will be called- UNIVERSITY database.
2. Moreover, primarily there will be 3 tables- TEACHERS, STUDENTS and STAFFS.
3. Now we will take a look at what personal records we should keep.

For teachers, what information do you really think should be kept?

- TITLE,
- NAME,
- DEPARTMENT,
- DESIGNATION,
- JOINING_DATE,
- DATE_OF_BIRTH,
- PHONE,
- E-MAIL,
- ADDRESS,
- NUMBER_OF_PUBLICATIONS.

For students, what information do you suggest should be kept?

- TITLE,
- NAME,
- ROLL,
- DEPARTMENT,
- YEAR,
- SEMESTER,
- DATE_OF_BIRTH,
- PHONE,
- E-MAIL,
- ADDRESS.

For staffs, we require at least this information-

- TITLE,
- NAME,
- DEPARTMENT,
- DESIGNATION,
- JOINING_DATE,
- DATE_OF_BIRTH,
- PHONE,
- E-MAIL,
- ADDRESS.

So, it is obvious now- we will have these *columns* in 3 tables respectively.

4. So, now think- student names can be the same, but they are distinguishable by their roll numbers. Is there anything for other two tables? How can you distinguish two teachers or two staffs with the same name? As a remedy, we can introduce 2 new information-

- TEACHER_ID in Teachers table and
- STAFF_ID in Staffs table.

These IDs are unique; no same ID is allocated to 2 different entries.

5. Now- take a look at the size of the tables- Huge, or short? If you want to update an entry, you may need to change values on every column! That is quite cumbersome job! So, we need to minimize tables. There are loads of benefits of doing this. We will find them in the later practical sessions. As we are not aware of these advantages yet, we will just try to implement it with our intuitions! Here are some of MY simple views- YOU CAN HAVE A DIFFERENT THINKING!

I would like to separate the data in following way-

- I will separate very personal data (ID/ ROLL, NAME) from others.
- I will separate information on date (JOINING_DATE, DATE_OF_BIRTH) from others.
- I will separate contact related info (ADDRESS, PHONE, E-MAIL) from others.

In a word- what I am trying to do is splitting up tables with small number of columns.

Up to this point, I have tables with columns as follows-

1. Teacher_Names (TEACHER_ID, TITLE, NAME)

2. Student_Names (ROLL, TITLE, NAME)
3. Staff_Names (STAFF_ID, TITLE, NAME)
4. Dates (ID, JOINING_DATE, DATE_OF_BIRTH)
5. Contacts (ID, ADDRESS, E-MAIL, PHONE)

We can have another table indicates which department the teacher/ student/ staff belongs to. So we can make the following table as well.

6. Departments (ID, DEPARTMENT)

Now, take a look at the first 3 tables we created- what columns are still needed to be covered? With them, try to make tables-

7. Designations (ID, DESIGNATION)
8. Publications (ID, NUMBER_OF_PUBLICATIONS)
9. YearSemesterRecords (ROLL, YEAR, SEMESTER)

So far, what I have seen, there is no other columns left to be added in our new tables. *It means, we made 9 tables from 3 table*s! Take a look- in all of our tables we have that unique identifier- ID/ ROLL!

Up to this point, this is a more reasonable design (I say it is designed effectively ☺). If you have any other suggestions, you can add it in your note book!

REMEMBER- DATABASE CAN WORK EVEN WITH A BAD (WORSE/ WORST) DATABASE DESIGN. BUT YOU WILL HAVE TO CODE LESS IF YOU DESIGN A DATABASE EFFECTIVELY!

Ok, now do something, try something on your own. This is your first step to design a database. So, there is nothing to be ashamed of. As days go by, you just take a look at this preliminary database. You will find out what to do it more efficient than now!

A Simple Case Study

A database is required to keep a software company's records. The company must keep it simple- it will only include some informational information, not more than that (for the sake of first day's lab!).

Information on management should be recorded. The company has two divisions- management and employee. In management division, there will be MD, DMD- this kind of people. Database should keep personal records like

- TITLE,
- NAME,
- POST,
- DATE_OF_BIRTH,
- JOINING_DATE,

- YEAR_OF_EXPERIENCE,
- PHONE, E-MAIL,
- ADDRESS,
- GENDER, etc.

About employees, similar kind of records can be kept.

- TITLE,
- NAME,
- POST,
- DATE_OF_BIRTH,
- JOINING_DATE,
- YEAR_OF_EXPERIENCE,
- PHONE,
- E-MAIL,
- ADDRESS,
- GENDER.
- One exception is NUMBER_OF_PROJECTS_COVERED.

The database also should contain records of its clients. Clients can be a company or individual. So, you decide if there should be two tables or just one will do the job.

- NAME_OF_COMPANY,
- GOVT_REGISTRATION_NUMBER,
- ADDRESS,
- WEB,
- E-MAIL,
- PHONE,
- FAX,
- NUMBER_OF_PROJECTS_SUPPLIED,
- TOTAL_COST,
- AMOUNT_PAID,
- AMOUNT_RECEIVABLE, etc.

Your task- just design it on your own, you judge it, show it to me if you want to, ask for my judgement! Good luck database designers! ☺