Introduction to Modern Application Development

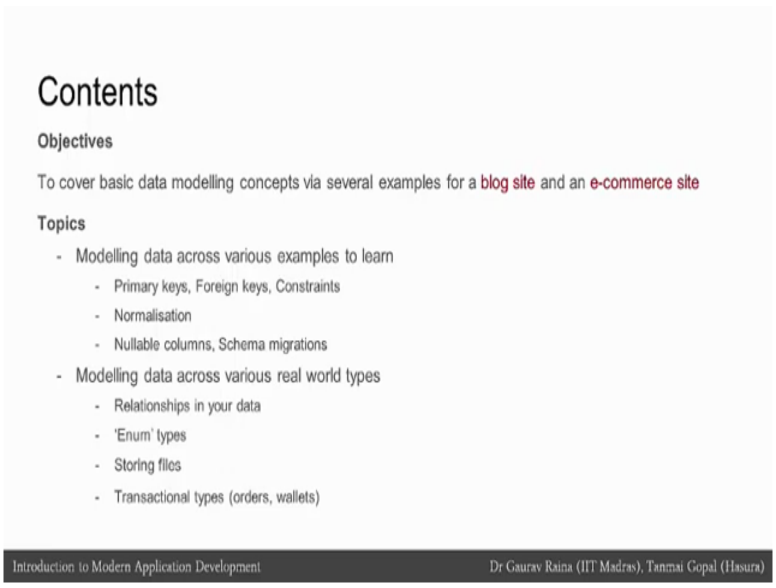
Dr. Gaurav Raina
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Module - 7
Lecture - 14
Data modelling & constraints

This module is going to be centered around extremely an important concept. And that concept is data modelling and constraints. Now, data modelling is an extremely integral part of a modern application development.

This module will introduce you to a few conceptual points, which would be very relevant in the space of data, its various modelling and the constraints that one would face.

(Refer Slide Time: 00:31)



Contents

Objectives

To cover basic data modelling concepts via several examples for a blog site and an e-commerce site

Topics

- Modelling data across various examples to learn
  - Primary keys, Foreign keys, Constraints
  - Normalisation
  - Nullable columns, Schema migrations
- Modelling data across various real world types
  - Relationships in your data
  - 'Enum' types
  - Storing files
  - Transactional types (orders, wallets)

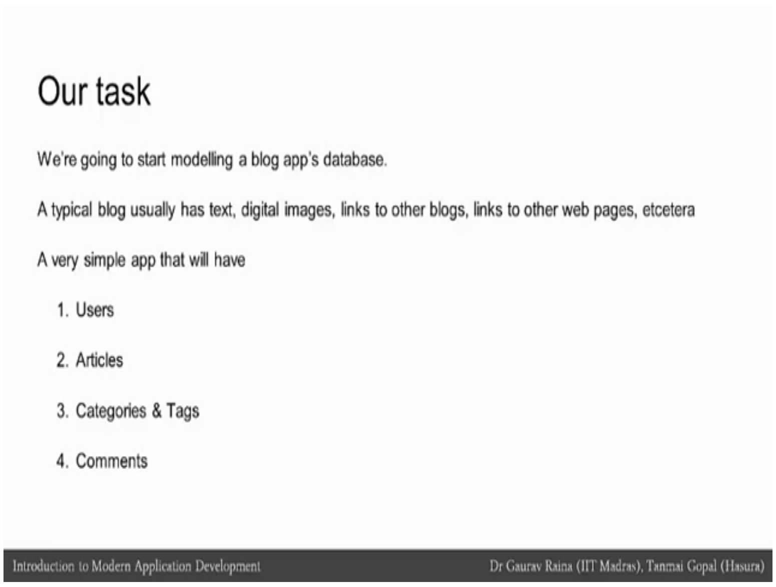
Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

The high level objective of this module is to cover basic data modelling concepts, as we said. But, what we are trying to do is actually get the points across through examples. In particular, we will be focusing on a blog site and an also an e-commerce site.

So, when you are looking at this space of data modelling, then there are few concepts that you need to be familiar with. The kinds of concepts are primary keys, foreign keys, constraints, normalization, nullable columns, schema migrations and so forth. So, one of the things to just note, which I just want to sort of highlight before we get into this module is that some of these concepts may be slightly conceptually tricky to absorb initially. But, do not worry about that.

The whole objective of this particular module is to expose you to some of these concepts. So, you should know that these concepts are relevant. And, certainly when we do the practicals, then the lot of these concepts should become much clearer to you. Now over and above, the above concepts, one also needs to pay particular attention to modelling data across real world types. So, you should also be; and will eventually become familiar with the relationships with in your data, enum types, storing files and transactional types.

(Refer Slide Time: 01:58)



Our task

We're going to start modelling a blog app's database.

A typical blog usually has text, digital images, links to other blogs, links to other web pages, etcetera

A very simple app that will have

1. Users
2. Articles
3. Categories & Tags
4. Comments

Introduction to Modern Application Development Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

So, our task is going to be the following. We got start modelling a blogs app's database. Now, a typical blog usually has a combination of text, digital images, links to other blogs, links to other web pages and so on and so forth. So, what will a very simple blog app have? It would have users, it would have articles, it would have categories and tags and it could have comments. Now let us look at each of these individually.

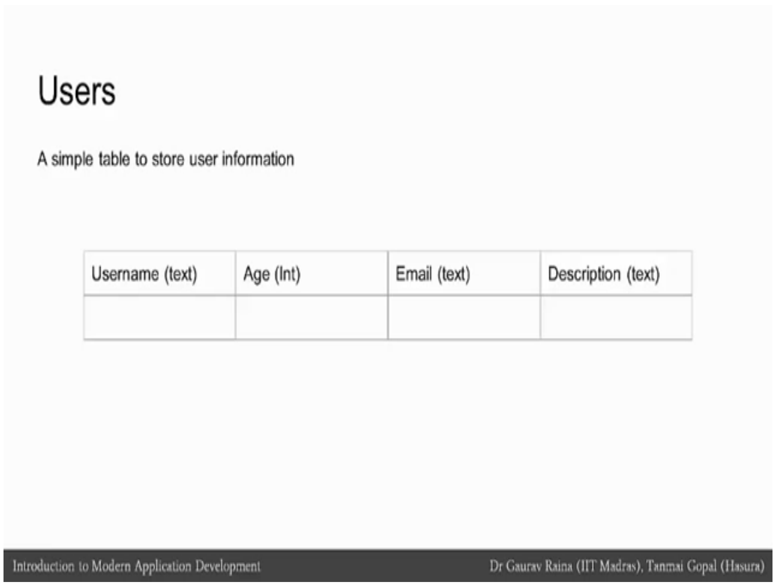
Now, users can log in and actually write articles. Now, each article will have only one author; each article will have one category, but perhaps many tags. And, each article can have many comments, posted by users of the application. Now, one of the things to be careful of or one of the things really to understand at this point of time is categories and tags.

Let us look at categories first. Now, a category is essentially there to help identify what your blog is really all about. So, think of these as sort of general topics or you can even think of it a sort of tables of contents for your sites. So, (Refer Time: 03:27). The essential idea is it is there to assist readers finding the right type of content on your site. When you are looking at tags; now, tags are meant to describe very specific details of your post. So, one way of thinking about tags is that think of them as your site's index words. They are essentially the micro data that you can use to micro categorize your content. So, let us take an example.

Let us assume you have a personal blog. And, in your personal blog you really want to write about your life. So, your categories may be something like food or music or travel or the books that you read and so on and so forth.

Now when you go out and eat something and you want to write a post about something that you ate, it will naturally fit in to the food category. And, in that food category you can add tags like pasta or a pizza or Italian food or whatever else that you actually ate under the food category. So, tags would be things like pasta and pizza and the category would be food.

(Refer Slide Time: 04:50)



Users

A simple table to store user information

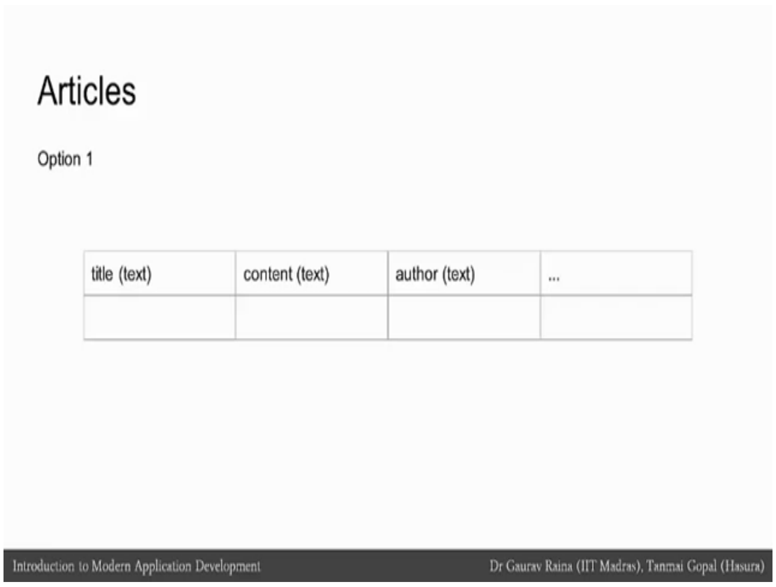
Username (text)	Age (Int)	Email (text)	Description (text)

Introduction to Modern Application Development

Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

Remember that users can actually log in and write articles. So, a very simple table to store user information would be the following. You have to use the name which is text, the age of the user in integer, email which is in text and description which is text.

(Refer Slide Time: 05:11)



Articles

Option 1

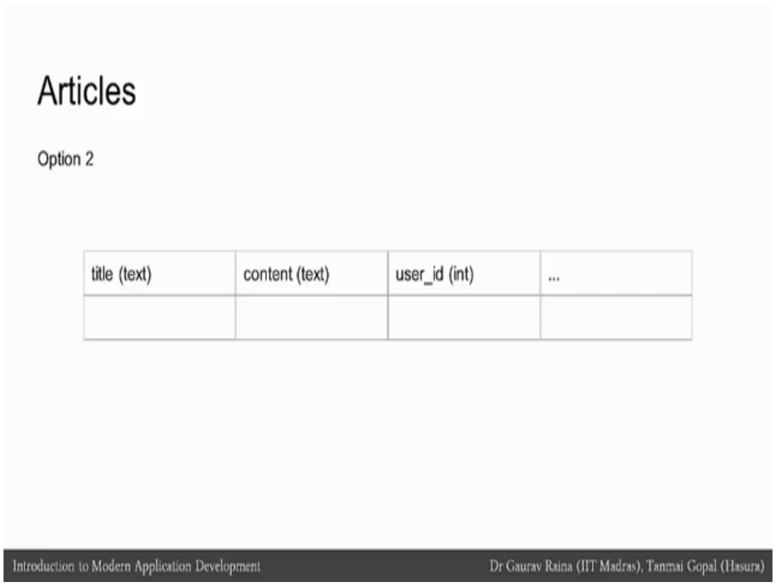
title (text)	content (text)	author (text)	...

Introduction to Modern Application Development Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

Now, when we get to the articles, here is one option for writing the information. That is, you have the title, the content and the author. Now, we can certainly do this. But, this may not be such a very good idea because the author name can be different from the user's name on the user table. Let me repeat that again.

The author's name can be different from the user's name on the user table. Now, this may become painful for the following reason; that suppose the user's name updates, then we wish to update all rows that have articles that are author, by the author whose name is actually changing. Now let us consider another option.

(Refer Slide Time: 06:06)



Articles

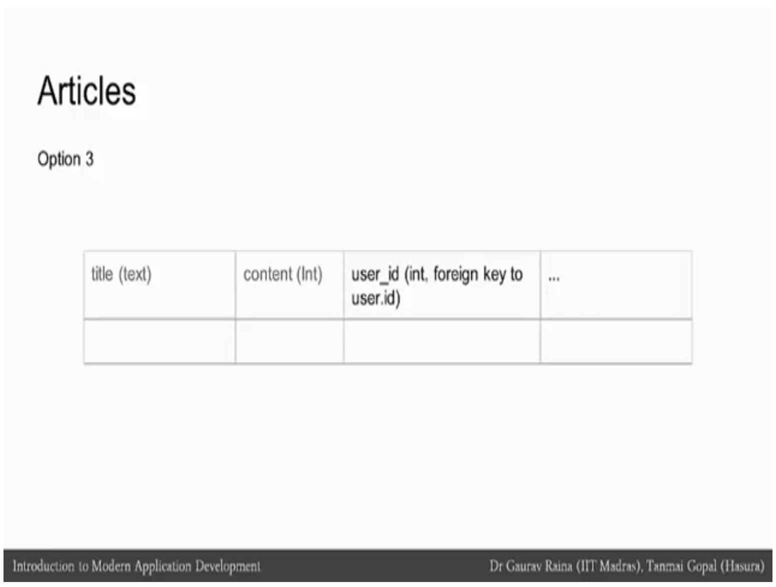
Option 2

title (text)	content (text)	user_id (int)	...

Introduction to Modern Application Development Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

Another option is you have title, content and the user i d. Now, this is better. But, this allows lots of other problems to create them, where if the user id does not actually exist. So, what if the user is actually deleted? Do the articles stay? Or should they also be deleted because the author is also deleted.

(Refer Slide Time: 06:39)



Articles

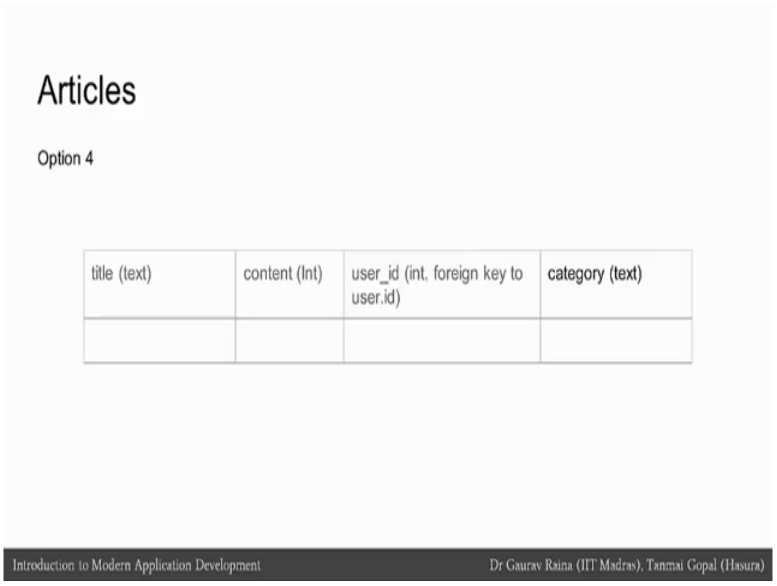
Option 3

title (text)	content (Int)	user_id (int, foreign key to user.id)	...

Introduction to Modern Application Development Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

Now let us look at option three; where we have title, content, user id. And, we are now introducing something known as a foreign key. Now, a foreign key is essentially a filed in one table that will you neatly identify a row of another table. It is a field in one table that uniquely identifies a row of another table. Let us put it in some simpler words. So, the foreign key is defined in a second table, but it refers to the primary key in the first table.

(Refer Slide Time: 07:24)



Articles

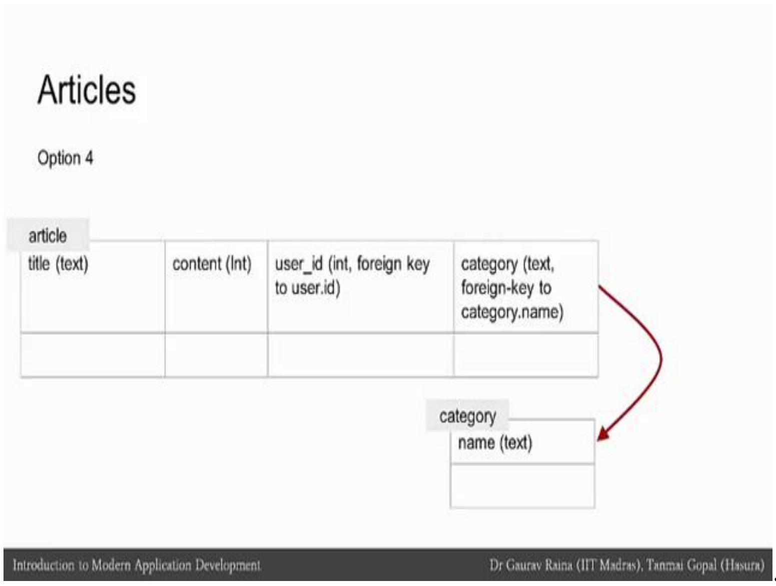
Option 4

title (text)	content (Int)	user_id (int, foreign key to user.id)	**category (text)**

Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

Now let us get on to categories. Remember if you were talking about a personal blog, when you are writing about your life, then sample categories where things like music, food, travel, books so on and so forth. Now, we have a category field and this can be a text field. So, here is the question. So, how can we guarantee that each article’s category is from a common set of categories? So, we do not want any new categories to be randomly created or specified along with an article.

(Refer Slide Time: 08:08)



Articles

Option 4

article

title (text)	content (Int)	user_id (int, foreign key to user.id)	category (text, foreign-key to category.name)

category

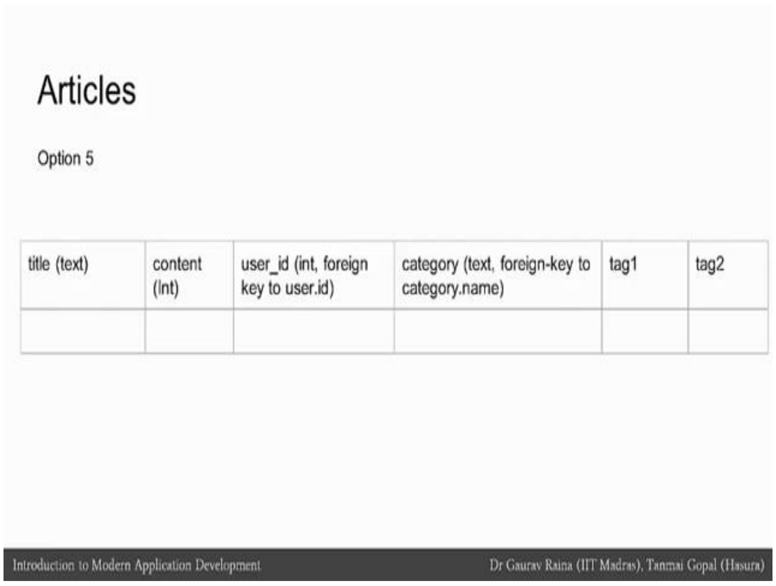
name (text)

Introduction to Modern Application Development
Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

.

Now, if you have a foreign key to category dot name, which translates to category name text. So, now essentially what happens is that random categories actually cannot be used or created in some random manner.

(Refer Slide Time: 08:30)



Articles

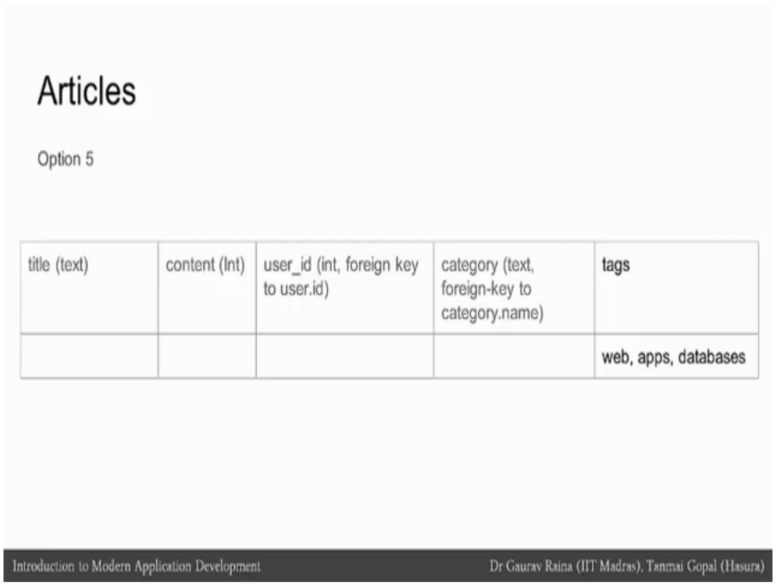
Option 5

title (text)	content (Int)	user_id (int, foreign key to user.id)	category (text, foreign-key to category.name)	tag1	tag2

Introduction to Modern Application Development
Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

And, now let us get on to tags. Remember again that in the category of food, if you went and ate something, you can add tags like pasta or pizza. So, we can add tag columns in this manner; tag 1, tag 2, tag 3, tag 4 so on and so forth. But, in the manner that has been described out here, it is limited and it is not very (Refer Time: 08:58).

(Refer Slide Time: 09:01)



Articles

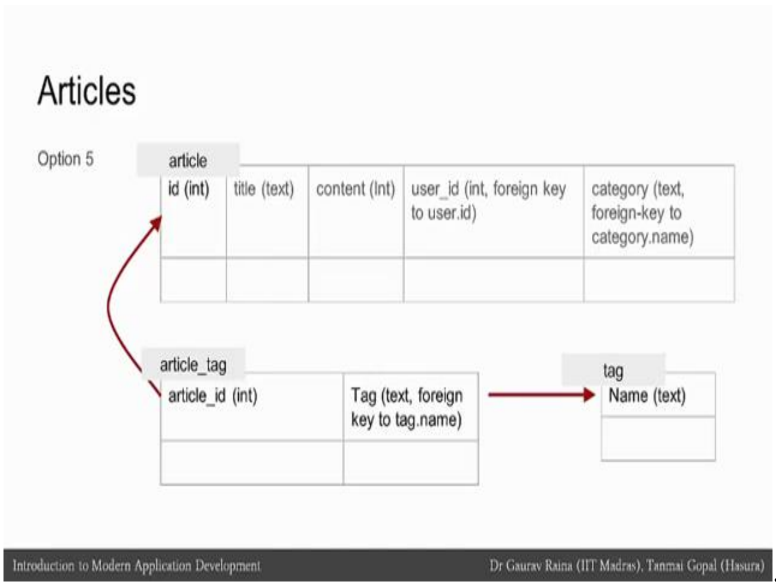
Option 5

title (text)	content (Int)	user_id (int, foreign key to user.id)	category (text, foreign-key to category.name)	tags
				web, apps, databases

Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

So, what we can do is we can add a tags column, which have comma separated tags. So, for example, you have web, apps, databases, so on and so forth. So, the question is how we can actually search for tags efficiently. Now, the problem is that you will have to read each row, then convert the tags from the comma separated values to tags and then match against each tag. So, you see that you will end up with the specific problem in your database. And that is that your database will not be able to index this specifically.

(Refer Slide Time: 09:39)



Articles

Option 5

article

id (int)	title (text)	content (Int)	user_id (int, foreign key to user.id)	category (text, foreign-key to category.name)

article_tag

article_id (int)	Tag (text, foreign key to tag.name)

tag

Name (text)

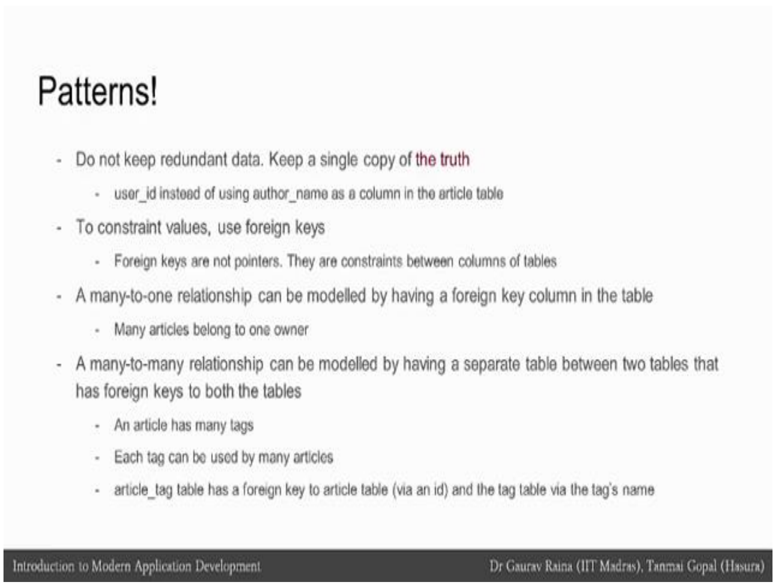
Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

.

So, you can do the following. You have article id integer which is article underscore tag, article underscore id, which is integer and tag. And, essentially what this does is we can actually find all articles for a particular tag very easily. So, let me repeat that again.

With this, we will be able to find all articles for a particular tag very easily. But, you end up with another problem. And, that is what if we want to tag? The tag to be from a subset of specific tags, so the question we want to really address is what if you want the tag to be from a subset of specific tags. And, in which case we do something very similar that we did to the category table. Remember what we have done to the category table. That is essentially what you do here as well. And, hence we will be able to look at the tag from a subset of specific tags.

(Refer Slide Time: 10:48)



Patterns!

- Do not keep redundant data. Keep a single copy of the truth
  - user_id instead of using author_name as a column in the article table
- To constraint values, use foreign keys
  - Foreign keys are not pointers. They are constraints between columns of tables
- A many-to-one relationship can be modelled by having a foreign key column in the table
  - Many articles belong to one owner
- A many-to-many relationship can be modelled by having a separate table between two tables that has foreign keys to both the tables
  - An article has many tags
  - Each tag can be used by many articles
  - article_tag table has a foreign key to article table (via an id) and the tag table via the tag's name

Introduction to Modern Application Development Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

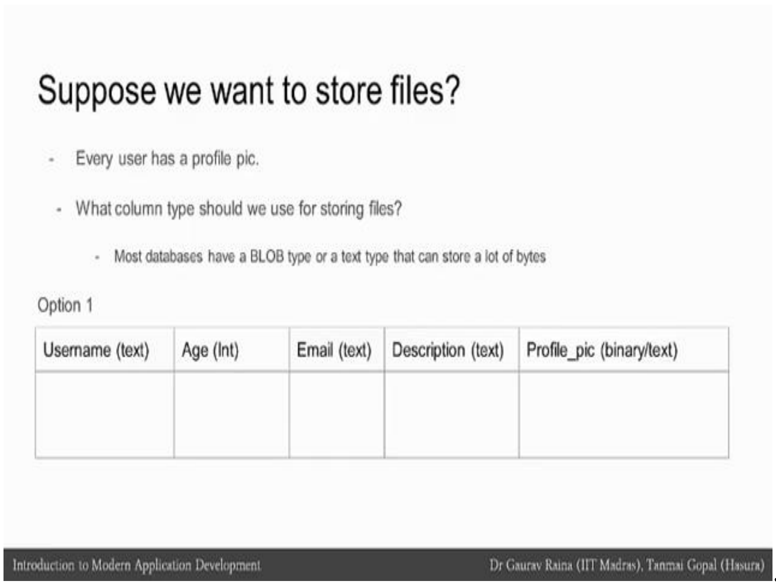
Now that we gone through this example, there are numerous patterns that seem to be coming up. The very first one is - do not keep redundant data, only keep a single copy of the truth. Now, this cannot be emphasized enough. And to emphasize it again, I repeat myself, do not keep redundant data. So, user underscore id, instead of using author underscore name as a column in the article table.

The second main point was that to constrain values, use foreign keys. Now, let us recall that a foreign key is a field in one table that you neatly identify a row of another table. So, note that foreign keys are not pointers. They are constraints between columns of tables.

The third point is that a many-to-one relationship can be modelled by having a foreign key column in the table. Say for example, many articles can belong to one owner. And finally, a many-to-many relationship can be easily modelled by having a separate table, between two tables that has foreign keys to both the tables.

For an example, an article has many tags. Each tag can be used by many articles. And, article underscore tag table has a foreign key to article table, via an id. And, the tag table, via the tag's name. So, these are four key patterns that will come up again and again and again. And, we should be aware of them.

(Refer Slide Time: 12:43)



Suppose we want to store files?

- Every user has a profile pic.
- What column type should we use for storing files?
    - Most databases have a BLOB type or a text type that can store a lot of bytes

Option 1

Username (text)	Age (Int)	Email (text)	Description (text)	Profile_pic (binary/text)

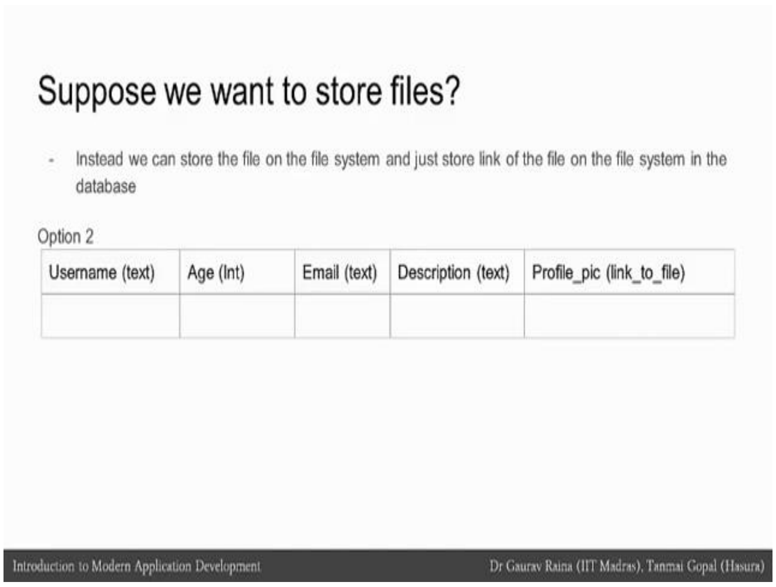
Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

.

And, now happens; if we want to actually store files. Now, it is quite common that every user will have a profile picture. So, what column should we use for storing files? Now, note that most databases have a BLOB type or a text type that can store a lot of bytes. So, here you will have profile underscore pic, which is binary slash text. But, this is actually a bad idea.

Now, think about it why it might be a bad idea? The reason is that storing the entire file in the database will actually end up causing serious performance issues as the database would load the file as well, which can be a lot of data. Every time we need to read or update the other fields of the row in the table. So, we have the entire profile pic there. Every time there is a read or an update, all of this data will be updated or moved.

(Refer Slide Time: 13:49)



Suppose we want to store files?

- Instead we can store the file on the file system and just store link of the file on the file system in the database

Option 2

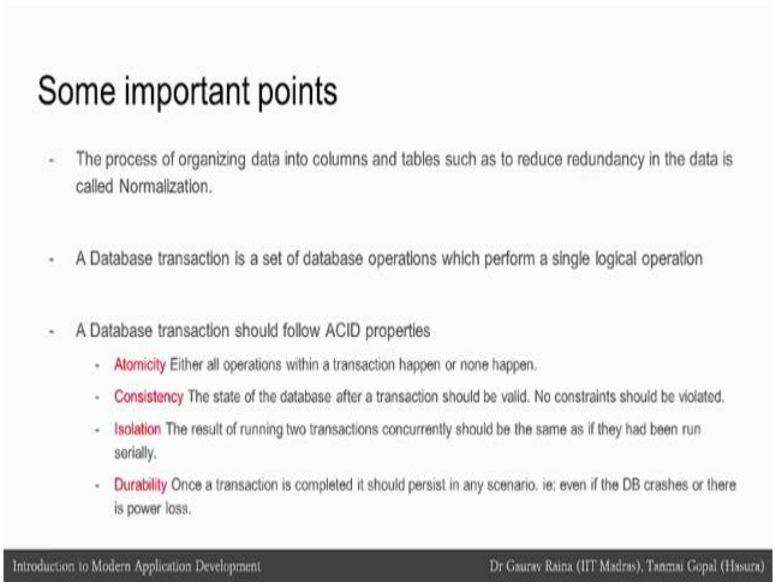
Username (text)	Age (Int)	Email (text)	Description (text)	Profile_pic (link_to_file)

Introduction to Modern Application Development Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

Instead, what we can do is we can store the file on the file system and just store the link of the file on the file system in the database. Essentially you have a profile underscore pic and then you have link to the file. But, now the thing is that the developer will have to make sure that the constraint, but (Refer Time: 14:16) the file link. And, the actual file is maintained as the database actually cannot help. If a file is deleted or its link is updated, this unfortunately cannot be handled by the database.

But, it is worthy to mention that the performance improvements are worth the trouble, because essentially if you do not do it this way, you will actually be loading the database, every single time there is a read or an update. So, what is the high level conclusion? High level conclusion is please do analyze the trade-offs very very carefully.

(Refer Slide Time: 15:01)



Some important points

- The process of organizing data into columns and tables such as to reduce redundancy in the data is called Normalization.
- A Database transaction is a set of database operations which perform a single logical operation
- A Database transaction should follow ACID properties
  - Atomicity Either all operations within a transaction happen or none happen.
  - Consistency The state of the database after a transaction should be valid. No constraints should be violated.
  - Isolation The result of running two transactions concurrently should be the same as if they had been run serially.
  - Durability Once a transaction is completed it should persist in any scenario. ie: even if the DB crashes or there is power loss.

Introduction to Modern Application Development — Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

Now, let us make some summary of some important points. The first is that the process of organizing data into columns and tables, such as to reduce redundancy in the data is called normalization. Second is that a database transaction is a set of database operations which performs a single logical operation. And finally, a database transaction should follow something known as ACID properties.

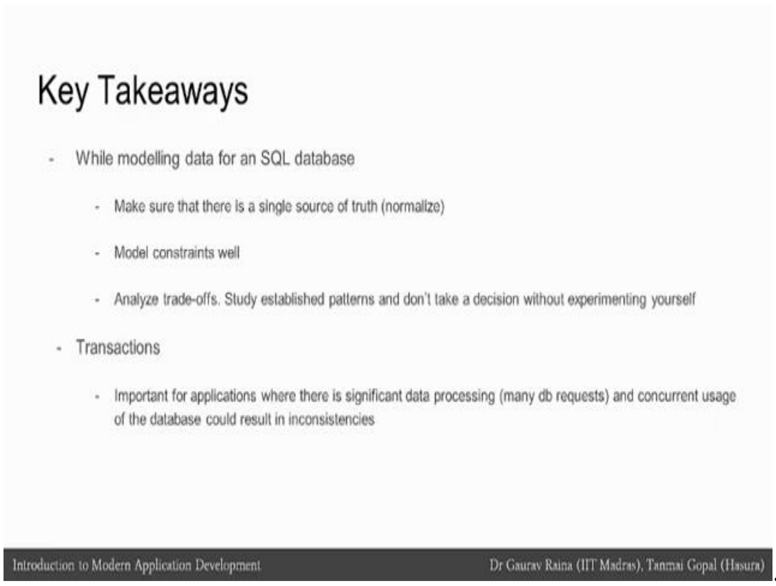
Atomicity, that is either, all operations within a transaction happen or none at all. Now, this can a make sense. You cannot have half the number of transaction happening. Either, all the transactions happened or none of the transactions happen.

Second is consistency. That is, the state of the database after transaction should be valid. What is that mean? That really means that no constraints that we have imposed should actually be violated.

The third is isolation. And, this is an important point which is that the result of running two transactions concurrently should be the same as if they had been run serially. This is self-explanatory. But, essentially what is it says is that if you were to run two transactions, more or less of the same time, then it should be the same as if you had run the first one and then the second one, after that.

And finally, of course the issue of durability, that is, once the transaction is actually completed, it should persist in any consumable scenario. For example, even if the database crashes or there is a power loss, the transaction should persist even if the database crashes or power loss.

(Refer Slide Time: 16:52)



Key Takeaways

- While modelling data for an SQL database
  - Make sure that there is a single source of truth (normalize)
  - Model constraints well
  - Analyze trade-offs. Study established patterns and don't take a decision without experimenting yourself
- Transactions
  - Important for applications where there is significant data processing (many db requests) and concurrent usage of the database could result in inconsistencies

Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

.

Now let us highlight some key takeaways for this particular module; to recall that our objective was to give you an introduction to basic data modelling concepts. (Refer Time: 17:08) while modelling data for an SQL database, the first thing to remember is to please make sure that there is only a single source of the truth; remove any redundancy in the data, i.e., normalize it.

The second is that, understand and model your constraints well. The third bit is analyze all your trade-offs. So, essentially what that means is study established patterns and do not repeat, do not take a decision, without actually experimenting yourself. Now, this is really important. This is very important from a learning perspective. This is extremely important from a design perspective and from an Engineering perspective.

And finally, a key takeaway in terms of transactions is that it is rather important for application, where there is a significant amount of data processing. That is, there are many database requests. And, concurrent usage of the database could result in inconsistencies. So, this provides now a summary to data modelling and constraints within databases.