How would I store a date that can be partial (i.e. just the year, maybe the month too) and output it later with the same specifity?

Asked 16 years, 3 months ago Modified 12 years, 7 months ago
Viewed 3k times

4

I want to let users specify a date that may or may not include a day and month (but will have at least the year.) The problem is when it is stored as a datetime in the DB; the missing day/month will be saved as default values and I'll lose the original format and meaning of the date.

My idea was to store the real format in a column as a string in addition to the datetime column. Then I could use the string column whenever I have to display the date and the datetime for everything else. The downside is an extra column for every date column in the table I want to display, and printing localized dates won't be as easy since I can't rely on the datetime value... I'll probably have to parse the string.

I'm hoping I've overlooked something and there might be an easier way.

(Note I'm using Rails if it matters for a solution.)

ruby-on-rails database-design datetime

Share

Improve this question

Follow

asked Sep 18, 2008 at 5:18

Zach
24.8k ● 9 ● 45 ● 50

11 Answers

Sorted by: Highest score (default)

9

As proposed by Jhenzie, create a bitmask to show which parts of the date have been specified. 1 = Year, 2 = Month, 4 = Day, 8 = Hour *(if you decide to get more specific)* and then store that into another field.

The only way that I could think of doing it *without* requiring extra columns in your table would be to use jhenzie's method of using a bitmask, and then store that bitmask into the seconds part of your datetime column.

Share Improve this answer

Follow

edited May 23, 2017 at 11:58

Community Bot
1 ● 1

answered Sep 18, 2008 at 5:34

nickf
546k ● 198 ● 658 ● 725

---

in your model only pay attention to the parts you care about. So you can store the entire date in your db, but you coalesce it before displaying it to the user.

3

Share Improve this answer

Follow

answered Sep 18, 2008 at 5:20

Aaron Jensen
6,060 ● 1 ● 32 ● 40

The trouble is, different users may have different levels of specificity - some only a year, some a month and day too. If it's too much trouble I might go with this, though. – Zach Sep 18, 2008 at 5:24

Ah, in that case I may just store it as multiple NULLABLE columns, one for year, month, day, then you'd have a FlexibleDate value object that'd interpret them.
– Aaron Jensen Sep 18, 2008 at 5:35

2

The additional column could simple be used for specifying what part of the date time has been specified

1 = day 2 = month 4 = year

so 3 is day and month, 6 is month and year, 7 is all three. its a simple int at that point

Share Improve this answer

Follow

answered Sep 18, 2008 at 5:25

user17256
29 ● 1

I feel this unnecessarily complicates things. Keep it simple.
– Aaron Jensen Sep 18, 2008 at 5:38

2

If you store a string, don't partially reinvent ISO 8601 standard which covers the case you describe and more:

http://en.wikipedia.org/wiki/ISO_8601

Share Improve this answer Follow

answered Sep 18, 2008 at 5:51

Paul Baclace

0

Is it really necessary to store it as a datetime at all ? If not stored it as a string 2008 or 2008-8 or 2008-8-1 - split the string on hyphens when you pull it out and you're able to establish how specific the original input was

Share Improve this answer Follow

answered Sep 18, 2008 at 5:23

southof40

Yes, I'll still need the benefits of a datetime column, such as ordering. – Zach Sep 18, 2008 at 5:35

0

I'd probably store the datetime and an additional "precision" column to determine how to output it. For output, the precision column can map to a column that contains the corresponding formatting string ("YYYY-mm", etc) or it can contain the formatting string itself.

Share Improve this answer

answered Sep 18, 2008 at 5:24

Follow

Mike Ivanov
153 ● 6

0

I don't know a lot about DB design, but I think a clean way to do it would be with boolean columns indicating if the user has input month and day (one column for each). Then, to save the given date, you would:

1. Store the date that the user input in a datetime column;
2. Set the boolean month column if the user has picked a month;
3. Set the boolean day column if the user has picked a day.

This way you know which parts of the datetime you can trust (i.e. what was input by the user).

Edit: it also would be much easier to understand than having an `int` *field with cryptic values!*

Share Improve this answer

Follow

answered Sep 18, 2008 at 5:28

André Chalella
14.1k ● 10 ● 57 ● 64

0

The informix database has this facility. When you define a date field you also specify a mask of the desired time & date attributes. Only these fields count when doing comparisons.

Share Improve this answer

Follow

answered Sep 18, 2008 at 5:37

Mike Thompson
6,738 ● 3 ● 33 ● 39

0

With varying levels of specificity, your best bet is to store them as simple nullable ints. Year, Month, Day. You can encapsulate the display logic in your presentation model or a Value Object in your domain.

Share Improve this answer

Follow

answered Sep 18, 2008 at 5:37

Aaron Jensen
6,060 ● 1 ● 32 ● 40

I have a feeling this would be troublesome in Rails, but otherwise a valid suggestion. – Zach Sep 18, 2008 at 5:53

0

Built-in time types represent an instant in time. You can use the built in types and create a column for precision (Year, Month, Day, Hour, Etc.) or you can create your own date structure and use nulls (or another invalid value) for empty portions.

Share Improve this answer

Follow

answered Sep 18, 2008 at 5:38

0

For ruby at least - you could use this gem - partial-date
https://github.com/58bits/partial-date

Share Improve this answer Follow

answered May 26, 2012 at 20:25

Blue Waters
725 ●1 ●6 ●20