Composite Primary key vs additional “ID” column?

41

11

If we had a table like this:

Books (pretend "ISBN" doesn't exist)

- Author
- Title
- Edition
- Year of publication
- Price

One could argue that {Author,Title,Edition} could be a candidate/primary key.

What determines whether the candidate/primary key should be {Author,Title,Edition} or whether an ID column should be used, with {Author,Title,Edition} a unique index/key constraint?

So is

- Author (PK)
- Title (PK)
- Edition (PK)
- Year of publication
- Price

better, or:

- ID (PK)
- Author
- Title
- Edition
- Year of publication

- Price

where {Author,Title,Edition} is an additional unique index/constraint?

sql sql-server database database-design

asked Jan 29 '13 at 17:09

user997112

10.4k 30 110 228

4 Poss dupe: stackoverflow.com/questions/1383062/composite-primary-key. Also read stackoverflow.com/questions/337503/… – MikeSmithDev Jan 29 '13 at 17:12

2 Search for "natural vs surrogate keys" or something similar. Many debates on this one. – Tim Lehner Jan 29 '13 at 17:12

2 This was discussed many times, for example: here, here and here. – Branko Dimitrijevic Jan 29 '13 at 17:36

4 Answers

38

Say that `{Author,Title,Edition}` uniquely identifies a book, then the following holds:

1. It is a (super)key -- uniquely identifies a tuple (row).
2. It is irreducible -- removing any of the columns does not make it a key any more.
3. It is a candidate key -- an irreducible key is a candidate key.

Now let's consider the ID (integer)

I can reason that the `Book` table key will show up in few other tables as a foreign key and also in few indexes. So, it will take quite a bit of space -- say three columns x 40 characters (or whatever...) -- in each of these tables plus in matching indexes.

In order to make these "other" tables and indexes smaller I can add a unique-integer-column to the `Book` table to be used as a key which will be referenced as a foreign key. Say something like:

```
alter table `Book` add `BookID` integer not null identity;
```

With `BookID` being (must be) unique too, the `Book` table now has two candidate keys.

Now I can select the `BookID` as a primary key.

```
alter table Book add constraint pk_Book primary key (BookID);
```

However, the `{Author,Title,Edition}` **must** stay a key (unique) in order to **prevent** something like this:

```
BookID  Author      Title           Edition
-----------------------------------------
  1      C.J.Date  Database Design     1
  2      C.J.Date  Database Design     1
```

To sum it up, adding the `BookID` -- and choosing it as the primary -- did not stop `{Author,Title,Edition}` being a (candidate) key. It still must have its own unique constraint and usually the matching index.

Also note that from the design point, this decision was done on the "physical level". In general, on the logical level of design, this `ID` does not exists -- it got introduced during the consideration of column sizes and indexes. So the physical schema was derived from the logical one. Depending on the DB size, RDBMS and hardware used, none of that size-reasoning may have measurable effect -- so using `{Author,Title,Edition}` as a PK may be perfectly good design -- until proven differently.

edited Jul 6 '17 at 10:06
App Work
10.4k 4 20 38

answered Jan 29 '13 at 20:04
Damir Sudarevic
19.2k 2 37 62

7

Another good reason for using the surrogate primary key scenario is if the uniqueness constraint should change in the future (say, ISBN needs to be added to make a book unique). Rekeying your data will be much easier.

answered Jan 29 '13 at 21:30
Scotty Boy
346 1 4

3

There are many articles related to this. The problems with composite key in your case:

1. hard to link books with other entities
2. Hard to edit them in a grid as most grids are not supporting composite keys (e.g. kendo grid, jqgrid)

3. You might misspell Author, Title, Edition

It would be also good to normalize your data and store just an ID to the author like (dasblinkenlight) suggested. Worst case scenario, he/she will change his/hers name (e.g. she get's married, and she likes her new name).

answered Jan 29 '13 at 17:29

Petrutiu Mihai
450 4 12

15

In general, you don't want the primary key to change value. This is why blind or surrogate primary keys are used.

Let's assume you created your Book table with Author as part of the primary key.

Suppose you found out after about a year that you misspelled "Ray Bradbury". Or even worse, you misspelled "Rachael Bloom". Just imagine how many database rows you would have to modify to correct the misspelling. Just imagine how many index references have to be changed.

However, if you have an Author table with a surrogate key, you only have to correct one row. No indexes have to be changed.

Finally, database table names are usually singular (Book), rather than plural (Books).

answered Jan 29 '13 at 17:18

Gilbert Le Blanc
38.8k 5 51 97

7 Disagree about singular vs. plural (a table contains a set of something (e.g. "books"), even if that set is just one row). But this is a completely different subjective argument and doesn't belong here IMHO. – Aaron Bertrand Jan 29 '13 at 17:19

When you say surrogate do you mean ID/ISBN is the one to use? – user997112 Jan 29 '13 at 17:29

I mean a primary key that has nothing to do with books at all. Generally, an integer that starts at zero and increments by one for every row added. I call them blind keys. – Gilbert Le Blanc Jan 29 '13 at 17:31

@AaronBertrand Singular is the accepted convention in ER modeling (e.g. search for "singular" in ERwin Methods Guide). Also, class names in OOP are singular, even though they can potentially be instantiated many times. But I do agree this is somewhat subjective and using a different convention is certainly not a cardinal sin... – Branko Dimitrijevic Jan 29 '13 at 17:31

3 I agree with Gilbert. I almost always add some kind of row identifier (usually an int identity column) even when it seems like there is something else that could be the key. Look at this - which is easier update users where username = 'jr33s22dt6uwwdrrws33ww' or update users where userid = 4 – Jeff B Apr 23 '13 at 11:53