How is Life for a Table in an Evolving Relational Schema? Birth, Death & Everything in Between

Panos Vassiliadis[1], Apostolos V. Zarras[1], and Ioannis Skoulis*[2]

[1] Dept. of Computer Science & Engineering, Univ. of Ioannina, Hellas
{pvassil, zarras}@cs.uoi.gr
[2] Opera Software, Oslo, Norway giskou@gmail.com

Abstract. In this paper, we study the version history of eight databases that are part of larger open source projects, and report on our observations on how evolution-related properties, like the possibility of deletion, or the amount of updates that a table undergoes, are related to observable table properties like the number of attributes or the time of birth of a table. Our findings indicate that (i) most tables live quiet lives; (ii) few top-changers adhere to a profile of long duration, early birth, medium schema size at birth; (iii) tables with large schemata or long duration are quite unlikely to be removed, and, (iv) early periods of the database life demonstrate a higher level of evolutionary activity compared to later ones.

Keywords: Schema evolution, patterns of change, design for evolution

1 Introduction

Databases evolve over time, both in terms of their contents, and, most importantly, in terms of their schema. Schema evolution affects all the applications surrounding a database, as changes in the database schema can turn the surrounding code to be syntactically or semantically invalid, resulting in runtime crashes, missing or incorrect data. Therefore, the understanding of the mechanics of schema evolution and the extraction of patterns and commonalities that govern this process is of great importance, as we can prepare in time for future maintenance actions and reduce both effort and costs.

However, in sharp contrast to traditional software, *the study of the mechanics of schema evolution is quite immature, as it includes only few case studies, each with limited insights into the topic* [8, 1, 5, 4, 10, 7]. The reason is quite simple, as the research community would find it very hard to obtain access to monitor database schemata for an in-depth study over a significant period of time. Fortunately, the emergence of open-source repositories (svn/ sourceforge / github) allows us to overcome this limitation, as we can gain access to the entire history of data-intensive project files, including the versions of the files with the database schema definition.

* work done while in the Univ. Ioannina

In this context, we embarked in the adventure of uncovering the internal mechanics of schema evolution, after having collected and analyzed a large number of database histories of open-source software projects. In [9], we have reported on our findings for the compatibility of database schema evolution with Lehman's laws and show that whereas the essence of Lehman's laws holds, the specific mechanics have important differences when it comes to schema evolution. In this paper, we come with a different contribution, that zooms into the details of the evolution of individual tables rather than entire relational database schemata and explore *how evolution-related properties, like the possibility of deletion, life duration, or the amount of updates that a table undergoes, are related to observable table properties like the number of attributes or the version of birth of a table*. Our guiding research questions and their answers follow.

Which tables are the ones that attract updates? Clearly, low-change tables dominate the landscape. In fact, when we studied the relation of life duration and amount of updates, we observed the *inverse Γ pattern* which states that updates are not proportional to longevity: with the exception of few long-lived, "active" tables, all other types of tables come with an amount of updates that is less than expected. These active tables with the larger amount of updates are long lived, frequently come from the first version of the database and, unexpectedly, they are not necessarily the ones with the largest schema size, but typically start as medium sized (although some of them, exactly due to their updates, eventually obtain a large number of attributes).

Which tables eventually survive and which ones are deleted? Our study has identified that there exist *"families"* of tables whose survival or removal is related to their characteristics.

- *Wide survivors*. The relation of schema size with duration revealed another interesting pattern, which we call *the Γ pattern*: "thin" tables, with small schema sizes, can have arbitrary durations, whereas "wide" tables, with larger schema sizes, last long.
- *Entry level removals*. The population of deleted tables is mostly located in a cluster of newly born, quickly removed, and with few or no updates; at the same time, we observe very low numbers of removed tables with medium or long durations. The schema size of deleted tables is typically small, too.
- *Old timers: time is on my side*. It is quite rare to see tables being removed at old age; although each data set comes with a few such cases, typically, the area of high duration is overwhelmingly inhabited by survivors!

We attribute the above phenomena to the "dependency magnet" nature of tables: the more dependent applications can be to their underlying tables, the less the chance of removal is. Large numbers of attributes, or large number of queries developed over time make both wide and old tables unattractive for removal.

Roadmap. The rest of this paper is structured as follows. In Section 2 we discuss related work. In Section 3 we present the experimental setup of our study. We present our findings in Sections 4 and 5. In Section 6, we discuss threats to validity. In Section 7, we conclude with a discussion on the practical implications of our findings and open issues. For more material, including links

to software and data, we indicate the web page of this paper (`http://www.cs.uoi.gr/~pvassil/publications/2015_ER/`) to the reader.

2 Related work

The first empirical study on the evolution of database schemas has been performed in the 90's [8]. The author monitored the database of a health management system for a period of 18 months. The results of the study showed that all the tables of the schema were affected and the schema had a 139% increase in size for tables and 274% for attributes. The author further observed a significant impact of the changes to the related queries. The second empirical study has been performed much later [1]. In this study, the authors analyzed the database back-end of MediaWiki, the software that powers Wikipedia. The results showed a 100% increase in schema size. However, 45% of changes did not affect the information capacity of the schema (but were rather index adjustments, documentation, etc.). The authors further provided a statistical study of lifetimes, change breakdown and version commits. This line of research was based on PRISM (recently re-engineered to PRISM++ [2]), a change management tool. Certain efforts investigated the evolution of databases, along with the applications that depend on them. In [10], the authors focused on 4 case studies. They performed a change frequency and timing analysis, which showed that the database schemas tend to stabilize over time. In [5], two case studies revealed that schemas and source code do not always evolve in sync. In [7], the authors investigated ten case studies. The results indicated that database schemas evolve frequently during the application lifecycle, with schema changes causing a significant amount of code level modifications.

Recently, we studied the applicability of Lehman's laws of software evolution in the case of database schemas [9]. In this study we found evidence that schemas grow so as to satisfy new requirements. However, this growth does not evolve linearly or monotonically, but with periods of calmness and short periods of focused maintenance activity. After a sufficient amount of changes, the schema size reaches a more mature level of stability.

Whereas previous work has mostly focused on the macroscopic study of the entire database schema, in this paper, we zoom into the lives of tables. We study the relationship of table duration, survival and update activity with table characteristics like schema size, time of birth etc. To the best of our knowledge, this is the first time that such findings appear in the related literature.

3 Data Compilation and History Extraction

In this section, we briefly present the eight datasets that we collected, sanitized, and studied using our open source SQL diff tool, Hecate. Those datasets include Content Management Systems (CMS's), Web Stores, along with Medical and Scientific storages (see Fig. 1).

Dataset	Versions	Lifetime	Tables @Start	Tables @End	Attributes @Start	Attributes @End	% commits with change
ATLAS Trigger	84	2 Y, 7 M, 2 D	56	73	709	858	82%
BioSQL	46	10 Y, 6 M, 19 D	21	28	74	129	63%
Coppermine	117	8 Y, 6 M, 2 D	8	22	87	169	50%
Ensembl	528	13 Y, 3 M, 15 D	17	75	75	486	60%
MediaWiki	322	8 Y, 10 M, 6 D	17	50	100	318	59%
OpenCart	164	4 Y, 4 M, 3 D	46	114	292	731	47%
phpBB	133	6 Y, 7 M, 10 D	61	65	611	565	82%
TYPO3	97	8 Y, 11 M, 0 D	10	23	122	414	76%

Fig. 1. The datasets we have used in our study [9]

In a nutshell, for each dataset we gathered as many schema versions (DDL files) as we could from their public source code repositories (cvs, svn, git). We have targeted only changes at the database part of the project as they were integrated in the trunk of the project. The files were collected during June 2013. For all of the projects, we focused on their release for MySQL (except ATLAS Trigger, available only for Oracle). The files were then processed by our tool, Hecate, that allows the accurate detection of (a) updates at the attribute-level, and specifically, attributes inserted, deleted, having an changed data type, or participation in a changed primary key, and, (b) changes at the table-level, with tables inserted and deleted, in a fully automated way. For lack of space, we refer the reader to [9] and its supporting web page.

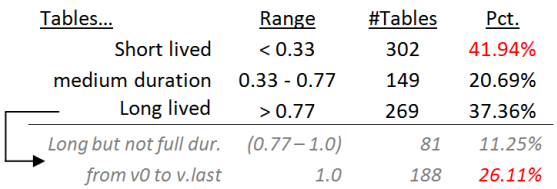
4 Table schema size, duration & updates

In this section, we study the relations between the tables' schema size, duration and updates.

Table schema size & duration

We have analyzed the lifetime duration of all the tables in all the studied datasets and we have observed that there is an interesting relation of durability with table schema size.

We computed the durations (in number of versions) for each table in each dataset. Then, again for each table, we produced a normalized measure of duration, by dividing the duration of the table by the duration of its database (more



Tables...	Range	#Tables	Pct.
Short lived	< 0.33	302	41.94%
medium duration	0.33 - 0.77	149	20.69%
Long lived	> 0.77	269	37.36%
Long but not full dur.	*(0.77 – 1.0)*	*81*	*11.25%*
from v0 to v.last	*1.0*	*188*	*26.11%*

Fig. 2. Distribution of tables with respect to their normalized duration

accurately, for the number of versions that we have monitored). This results in all tables having a normalized duration in the range of $(0 \ldots 1]$. Then, in an attempt to simplify intuition, we decided to classify tables in three categories: (i) short-lived (including both the ones who were removed from the system at some point and the ones whose short duration is due to their late appearance), (ii) tables of medium duration and (iii) long lived tables. To avoid manually specifying the limits for each of these categories, we used a k-means clustering [3] that, based on the normalized duration of the tables, split the data set in three clusters, and specifically at the values 0.33 and 0.77. The details of the breakdown appear in Fig. 2, where we devote one line per category, along with the breakdown of the long-lived category in two subclasses: (a) tables that live long, but not throughout the entire lifetime of the database and (b) tables that live from the very first till the last version that we have monitored (and thus, have a normalized duration equal to 1.0).

One of the fascinating revelations of this measurement was that there is a 26.11% fraction of tables that appeared in the beginning of the database and survived until the end. In fact, if a table is long-lived, there is a 70% chance (188 over 269 occasions) that it has appeared in the beginning of the database.

Fig. 3 gives the statistical distribution of the average table schema size (i.e., the average number of attributes a table has throughout its lifetime). Typically, the average table schema size has a very small deviation, as tables do not change largely. Thus, the average size is pretty close to the respective max and min size for the studied tables. *On average, half of the tables (approx. 47%) are small tables with less than 5 attributes. The tables with 5 to 10 attributes are approximately one third of the tables' population and the wide tables with more than 10 attributes are approximately 17% of the tables.* The first category has quite a few outliers, both high and low (and a large standard deviation of 20%), whereas the two others do not really oscillate too much (with standard deviations of 14 and 13%, respectively). Interestingly, the datasets with less evolutionary activity (atlas, typo3 and biosql) are the ones concentrating outlier values.

The first column of Fig. 4 gives the relation of a table's schema size (measured as the number of attributes of a table's schema at its birth and depicted in the x-axis of each of the scatterplots) with duration (y-axis in the scatterplots), in three characteristic data sets, Atlas, Coppermine and Mediawiki (for lack of space, it is impossible to show all datasets). We observe a phenomenon that

Pct of tables with num. of attributes …			
	<5	5-10	>10
atlas	10,23%	68,18%	21,59%
biosql	75,56%	24,44%	0,00%
coppermine	52,17%	30,43%	17,39%
ensembl	54,84%	38,06%	7,10%
mediawiki	61,97%	19,72%	18,31%
phpbb	40,00%	44,29%	15,71%
typo3	21,88%	31,25%	46,88%
opencart	57,20%	33,05%	9,75%
Average	46,73%	36,18%	17,09%

Fig. 3. Distribution of tables with respect to their average schema size

we call Γ *pattern*: *tables with small schema sizes can have arbitrary durations, whereas tables with larger schema sizes last long.*

Plainly put, the facts that the first column of Fig. 4 vividly demonstrates are as follows. There is a large majority of tables, in all datasets, whose size is between 0-10 attributes. We have visualized data points with transparency in the figure; therefore, areas of intense color mean that they are overpopulated with data points that fall on the same x,y coordinates. Many of the data sets are limited to really small sizes: biosql tables do not exceed 10 attributes, ensembl tables do not exceed 20 attributes and atlas, mediawiki and coppermine have one or two tables "wider" than 20 attributes (btw., observe the outlier table of 266 attributes in atlas, too). Due to their large percentage, small tables dominate the area near the beginning of the axes in the figure. What is interesting however, is that *their small size does not determine their duration*: this is depicted as a distribution of data points in parallel with the y-axis for all the small sizes of a table's schema. On the other hand, we observe that *whenever a table exceeds the critical value of 10 attributes in its schema, its chances of surviving are high. In fact, we observe that in most cases the "wide" tables are created early on and are not deleted afterwards.* We conjecture that the explanation for the "if you 're wide, you survive" part of the Γ pattern is due to the impact a table deletion has: the wider a table, the higher a chance it acts as a fact table, frequently accessed from the queries in the applications surrounding the database. Thus, its removal incurs high maintenance costs, making application developers and administrators rather reluctant to remove it.

Table schema size & updates

If we observe the middle column of Fig. 4, we can see the relation of a table's schema size at its birth (depicted in the x-axis of each of the scatterplots) with the amount of change the table undergoes (y-axis in the scatterplots). There are two main clusters in the plots: (*a) a large, dense cluster close to the beginning of the axes, denoting small size and small amount of change, and, (b) a sparse set*

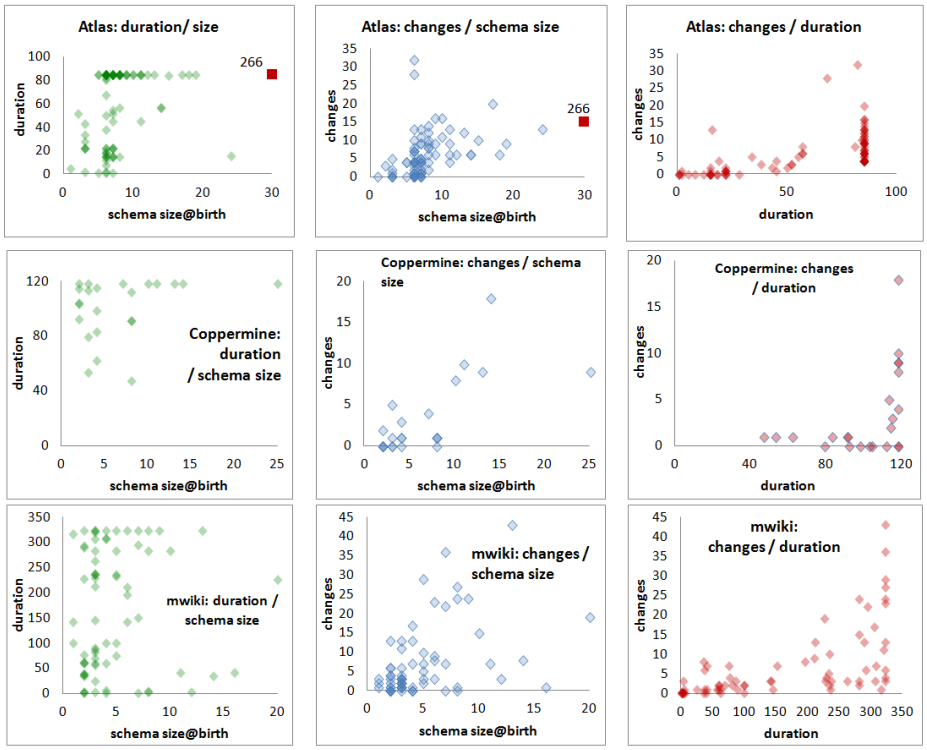


Fig. 4. Correlation of schema size (number of attributes), duration (over versions) and change (total number of changes) of all the tables in three datasets

of outliers, broken in two subcategories: (b1) medium schema size tables typically demonstrating medium to large amounts of changes and (b2) "wide" tables with large schema sizes demonstrating small to medium amount of change. We refer to this distribution as the *comet pattern.*

The first "nucleus" cluster is typically contained within a box of size 10×10 (i.e., no more than 10 attributes typically result in no more than 10 changes). This is attributed to the small pace of change that tables undergo (cf. Sec. 5), resulting in small probabilities of change for small tables. At the same time, in most of the datasets, *the tables with the largest amount of change are not necessarily the largest ones in terms of attributes, but tables whose schema is on average one standard deviation above the mean.* Typically, medium sized tables demonstrate all kinds of change behaviour as they cover the entire y-axis, whereas the (few) tables with large schema size demonstrate observable change activity (i.e., not zero or small), which is found around the middle of the y-axis of the plot in many cases. An extra observation, not depicted in the figure for lack of space, occurs when we observe the average schema size instead of the schema size at birth. The charts look quite alike with small differences, but in

several of the datasets we see tables with medium-sized schema obtaining a large schema via a large number of updates. In other words, *if a table is already wide, there is rarely a tendency for large growth; however, it is more often that a table of medium schema size is augmented a lot to carry more information.*

Table duration & updates

The last column of Fig. 4 demonstrates the relation of a table's duration (depicted in the x-axis of each of the scatterplots) with the amount of change the table undergoes (y-axis in the scatterplots). We observe a phenomenon that we call the *inverse Γ pattern*: *tables with small duration undergo small change, tables with medium duration undergo small or medium change, and, long-lived tables demonstrate all kinds of change behavior.*

The vast majority of tables have "calm" lives, without too much change activity; therefore, there is a high chance that a table will be low in the y-axis of the plot. In detail, all three scatterplots of the last column of Fig. 4 present three clusters (omnipresent in all eight datasets that we have studied). The absence of extremities means that a short lifetime cannot really produce anything but small (typically close to zero) change; medium duration has some higher chance of producing change in the range of 5 to 10 changes; and high amounts of change are only found in tables with long lives. Still, instead of a full triangle, the chart demonstrates mainly an inverse Γ: there is a striking scarcity of tables with mid-sized durations and mid-range change that would fill the interior of the triangle (visual clue: color intensity in our charts is an indicator of population density, due to overlapping semi-transparent points). The majority of tables having quiet, calm lives, pushes the triangle to become an inverse Γ.

5 Table birth, death & updates

Having examined how the schema size, duration and total amount of update of a table can be related, we further investigate how these measures are also related with table birth and death. One related question that has been puzzling us, concerns the relationship of update activity with the possibility of removal. As it is theoretically possible that a table with intense change activity has a short life, we introduced an extra parameter to the problem that assesses change activity. So, we define the *Average Transitional Update (ATU)* as the fraction of the sum of updates that a table undergoes throughout its life over its duration. Equivalently, one can think of this measure as *the average number of schema updates of a table per transition.* Thus, the average transitional update of a table practically measures how active or rigid its life has been by normalizing the total volume of updates over its duration.

We have assessed the average transitional update of all the tables in all datasets and we have studied its relation to the rest of the characteristics of the tables (schema size, version of birth and death, etc). To address the *survival vs death* separation task, we will employ the following terminology:

Table distribution (pct of tables) wrt their avg transitional update rate

		DIED				SURVIVED				*Aggregate per update type*		
	#tables	No change	Quiet (0-0.1)	Active (>0.1)	Total	No change	Quiet (0-0.1)	Active (>0.1)	Total	No change	Quiet (0-0.1)	Active (>0.1)
atlas	88	8%	7%	2%	**17%**	13%	42%	**28%**	**83%**	20%	49%	**31%**
biosql	45	20%	13%	4%	**38%**	16%	16%	**31%**	**62%**	36%	29%	**36%**
phpbb	70	0%	3%	4%	**7%**	50%	31%	**11%**	**93%**	50%	34%	**16%**
typo3	32	16%	6%	6%	**28%**	22%	34%	**16%**	**72%**	38%	41%	**22%**
coppermine	23	4%	0%	0%	**4%**	30%	57%	**9%**	**96%**	35%	**57%**	**9%**
ensembl	155	24%	20%	8%	**52%**	6%	37%	**5%**	**48%**	30%	**57%**	**13%**
mwiki	71	14%	13%	3%	**30%**	3%	63%	**4%**	**70%**	17%	**76%**	**7%**
opencart*	128	9%	2%	0%	**11%**	42%	44%	**3%**	**89%**	51%	46%	**3%**

Fig. 5. Table classification concerning the average amount of updates per version for all data sets (∗Opencart reports only on the tables born after transition 17)

- *survivor* is a table that was present in the last version of the database that we examine, and,
- *non-survivor* refers to a table that was eventually eliminated from the database.

We also discriminate the tables with respect to their update profile as follows:

- *Active tables* or *top-changers* are the ones with (a) a high average transitional update, exceeding the empirically set threshold of 0.1, and, (b) a non-trivial volume of updates, exceeding the also empirically set limit of 5 updates.
- *Rigid tables* are the ones with no change at all; we will also refer to the ones that did not survive as *sudden deaths*, as they were removed without any previous change to their schema.
- *Quiet tables* are the rest of the tables, with very few updates or average transitional update less than 0.1. For the tests we made, discriminating further within this category did not provide any further insights.

Observe Fig. 5 that depicts how tables are grouped in categories for all the datasets. We group measurements separately for (a) non-survivors, (b) survivors, and (c) overall. Within each of these groups, apart from the overall statistics, we also provide the breakdown for tables with (a) no changes, (b) quiet change, and (c) active change. Each cell reports on the percentage of tables of a data set that fall within the respective subcategory. We have slightly manipulated a single dataset, for the sake of clarity. OpenCart has an extreme case of tables renamings, involving (a) the entire schema at transitions 17 and 18 and (b) massive renamings in transitions 23 and 35. As these massive renamings involved 108 of the 236 tables of the monitored history, we decided to exclude the tables that were dead before transition 23 from our study, so that their change does not overwhelm the statistics. For example, after the aforementioned short period, only 14 tables were removed in the subsequent 130 revisions.

Clearly, quiet tables dominate the landscape. If we observe the non-survivors, the percentage of tables in each of the categories is typically dropping as the probability of change rises: *sudden deaths are the most frequent, quiet non-survivors come second, and third, a very small percentage of tables in the vicinity of 0-6%*

dies after some intense activity. In the case of the survivors, the percentage of rigid tables is often substantial, but, with the exception of phpbb, the landscape is dominated by quiet tables, typically surpassing 30% of tables and reaching even to 60% of the tables for a couple of data sets. Things become quite interesting if one focuses on the more mature data sets (coppermine, ensemble, mediawiki, opencart) whose profile shows some commonalities: in contrast to the rest of the datasets, active survivors are a small minority (approx. 5%) of the population, and similarly, active over both survivors and non-survivors ranges between 3-13%. So, *as the database matures, the percentage of active tables drops* (as already mentioned, evolution activity seems to calm down). It is also noteworthy that *despite similarities, and despite the obvious fact that at least two thirds of the tables (in the mature databases: 9 out of 10) lead quiet lives, the internal breakdown of updates obliges us to admit that each database (and developer community) comes with its own update profile.*

In Figure 6 we present the most interesting insights from the Mediawiki data set. We have chosen Mediawiki as a fairly representative dataset, as it comes with both a non-negligible amount of removed tables, a large duration, and, a fairly large number of tables (72). Our findings are discussed below.

Table birth, duration & updates

- *Surviving top-changers live long.* Observe the top chart of Figure 6: the vertical line of the *inverse* Γ in long durations, as well as its nearby region are being populated by survivors; the vertical line demonstrates that several of the top-changers are very long lived. Remember that the top-changers are the ones with the highest average transitional update, i.e., their update is normalized over their duration. This is quite different from the total amount of update that is depicted in the y-axis of the chart: in other words, it is theoretically possible (and in fact, there is indeed such a case in the Mediawiki example, found close to the beginning of the axes) that a short lived table with relatively low total updates is a top-changer. In practice, however, longevity and high average transitional update are closely related.
- Concerning the top-changers, there is an interesting correlation of high update activity, overall change, duration and birthdate: *most (although not all) top-changers are born early, live long, have high average transitional update* and, consequently a large amount of total update. Of course, there exist tables born early who survive and have lower update activity and top-changers that violate this rule; however, in all data sets, it is quite uncommon to observe top-changers outside the "box" or early birth and long duration.

Table death, duration & updates

- There is a large concentration of the *deleted tables in a cluster of tables that are newly born, quickly removed, with few or no updates.* Few deleted tables demonstrate either late birth, or long durations, or high average transitional update, or large number of updates (Mediawiki shows just a couple of them

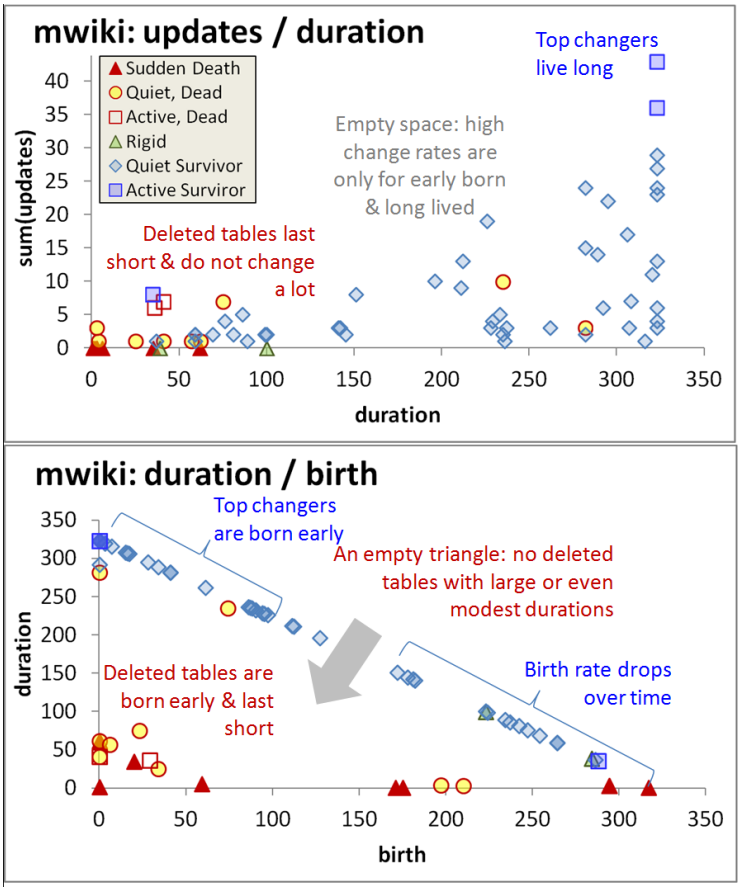


Fig. 6. Update profile with respect to survival-vs-death injected in the study of total change over duration (top) and duration over birthday (bottom)

in each of the first two categories). In the triangle formed in the bottom figure by the two axes and the line of survivors, the diagonal of survivors is expected: if you are a survivor, the earlier you have been born, the more you live. *The really surprising revelation of the triangle, however, is that (a) the diagonal is almost exclusively made from survivors, and, (b) the triangle is mainly hollow*! Theoretically, it is absolutely legitimate for non-survivors to have long durations and to be born in the middle of the database lifetime: if there was not a pattern of attraction of non-survivors to the beginning of the axes we could legitimately see non-survivors in the area of the diagonal and, of course, a uniform spread in the interior of the triangle. In fact, in the case of Mediawiki, we do see a couple (but just a couple) of such cases for both the aforementioned possibilities.

– *It is quite rare to see tables being removed at old age; although each data set comes with a few such cases, typically, the area of high duration is overwhelmingly inhabited by survivors!* See the upper part of Fig. 6 for the Mediawiki data set, showing just a couple of deleted tables with high durations. We believe (although cannot prove) that this has to do with the cost of removing a table after some time: applications have been built around it and the maintenance cost to the surrounding queries shrinks the possibility of removal.

Exceptions to the aforementioned observations do exist. In the case of Typo3, the diagonal is closely followed by a set of medium-change tables. This is mainly attributed to the fact that there is a short period of 5 transitions, around transition 70 during which almost all table deletions take place. Opencart has the characteristic of massive renamings that removed all the old tables until transition 34. After that, there are a few targeted restructurings of sudden deaths. Ensembl is the only data set with a heavy deletion oriented character (52% of tables removed). In the case of Ensembl, although the area close to the diagonal has just a couple of deleted tables, there are two differences: (a) the attraction to the beginning of the axes last quite longer than every other data set (a box of 135x135) and (b) there are several tables born around the middle of life of the data set that eventually die, having been fairly active and quite long-lived (approximately 200 transitions).

6 Threats to validity

We stress that our study has been performed on databases being part of open-source software. This class of software comes with a specific open modus operandi in terms of committing changes and distributing maintenance work. Thus, we should be very careful to not overgeneralize findings to proprietary databases. We would also like to stress that we work only with changes at the logical schema level (and ignore physical-level changes like index creation or change of storage engine). As another concern, one could possibly argue that application areas affect the way databases evolve (e.g., CMS's can behave differently than scientific databases). Although this is possible in general, in this paper we have focused on patterns that we found common in all data sets. Overall, we are quite confident with our results concerning this area, as we have covered a large number of databases, from a variety of domains that seem to give consistent answers to our research questions (behaviour deviations are mostly related to the maturity of the database and not to its application area).

7 Practical exploitation & open issues

Assume you are the chief architect of a large information system. Evolution is inevitable, maintenance takes up to 70% of resources that are always limited, time is pressing and you have to keep the system up to date, correct, without

failures and satisfactory for the users. In this section, we relate the key findings of our study with *guidelines on designing and building both tables and applications that access them, so as to sustain the evolution of the database part gracefully.*

Life and death of tables: only the thin die young, all the wide ones seem to live for ever. The Γ pattern says: "Large-schema tables typically survive". Yet, more than half of the tables are of short schema sizes and up to 80% of tables have less than 10 attributes, and in fact, it is the short-sized tables' category being the one with short durations and table deletions. The deletions of these "narrow" tables typically take place early in the lifetime of the project, either due to deletion or due to renaming (which is equivalent from the point of view of the applications: they crash in both cases). To address change, we strongly denounce table denormalization as a solution that reduces the number of potentially evolution-prone, thin tables by introducing few, artificially wide, reference tables. Rather, we suggest the usage of views as an *evolution-buffering, "API-like" mechanism* between applications and databases.

> *Mask your applications from tables prone to change (e.g., tables with few attributes) as much as possible, especially at the early versions of the database where most of the deletions take place. To this end, use views as a buffering mechanism between the database and the applications that masks change in several evolution scenarios like renamings, table splitting or merging, etc.*

Everything in between: quiet growth. Due to the large impact that their update induces, most tables lead quiet lives with few updates (remember: additions and deletions of attributes, data type and key changes). Medium-rated change is scarcely present too: in fact, the *inverse* Γ pattern states that updates are not proportional to longevity, but rather, only few top-changer tables deviate from a quiet, low-change profile. Top-changers tables are long lived, frequently come from the first version of the database, can have large number of updates (both in absolute terms and as a normalized measure over their duration) and, unexpectedly, are not necessarily the wider ones, but frequently their schema is typically medium sized. Although each database comes with its own idiosyncracy, we still can coarsely say that early stages of the database life are more "active" in terms of births, deaths and updates, whereas, later, activity is less "hot", including table additions (which seem to happen throughout the entire timeline), and deletions and updates that become more concentrated and focused. The overall result, also due to the outnumbering of table deletions by table additions is that the schema of the database expands over time.

> *Plan for a quiet expansion of the schema! The database is augmented with new tables all the time; at the same time, although most tables lead quiet lives and do not grow much, some top-changers are "change attractors". To address the database expansion, try to systematically use metadata rich facilities involving the interdependencies of applications and tables to be able to locate where the code should be maintained for added, deleted or modified attributes and tables.*

Open issues. Possibilities for follow-up work are immense. For lack of space, we restrict ourselves to a few points. First, there is always the ambitious scientific goal of having some "weather forecast" for the forthcoming changes of a database (we should, however, warn younger readers on the level of risk this research encompasses). Second, one can deal with the possibility of capturing more types of changes (like renamings, normalization actions, etc) that have not been part of our fully automated mechanism for transition extraction. Third, one could also work on the arguably feasible engineering goal of having flexible structures for gluing applications to evolving database schemata, thus minimizing application dependency and evolution impact (see [6] for our take on the problem).

Acknowledgments. This work was partially supported from the European Community's FP7/2007-2013 under grant agreement number 257178 (project CHOReOS). We would like to thank the reviewers of the paper for helpful comments and suggestions for solidifying our work.

References

1. Curino, C., Moon, H.J., Tanca, L., Zaniolo, C.: Schema evolution in wikipedia: toward a web information system benchmark. In: Proceedings of ICEIS 2008. Citeseer (2008)
2. Curino, C., Moon, H.J., Deutsch, A., Zaniolo, C.: Automating the database schema evolution process. VLDB J. 22(1), 73–98 (2013)
3. Dunham, M.H.: Data Mining: Introductory and Advanced Topics. Prentice-Hall (2002)
4. Hartung, M., Terwilliger, J.F., Rahm, E.: Schema Matching and Mapping, chap. Recent Advances in Schema and Ontology Evolution, pp. 149–190. Springer (2011)
5. Lin, D.Y., Neamtiu, I.: Collateral evolution of applications and databases. In: Proceedings of the Joint International and Annual ERCIM Workshops on Principles of Software Evolution (IWPSE) and Software Evolution (Evol) Workshops. pp. 31–40. IWPSE-Evol '09 (2009)
6. Manousis, P., Vassiliadis, P., Papastefanatos, G.: Automating the adaptation of evolving data-intensive ecosystems. In: Proceedings of 32th International Conference on Conceptual Modeling (ER 2013), Hong-Kong, China, November 11-13, 2013. pp. 182–196
7. Qiu, D., Li, B., Su, Z.: An empirical analysis of the co-evolution of schema and code in database applications. In: Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering. pp. 125–135. ESEC/FSE 2013 (2013)
8. Sjøberg, D.: Quantifying schema evolution. Information and Software Technology 35(1), 35–44 (1993)
9. Skoulis, I., Vassiliadis, P., Zarras, A.: Open-source databases: Within, outside, or beyond Lehman's laws of software evolution? In: Proceedings of 26th International Conference on Advanced Information Systems Engineering - CAiSE 2014 (2014), available at `http://www.cs.uoi.gr/~pvassil/publications/2014_CAiSE/`
10. Wu, S., Neamtiu, I.: Schema evolution analysis for embedded databases. In: Proceedings of the 2011 IEEE 27th International Conference on Data Engineering Workshops. pp. 151–156. ICDEW '11 (2011)