University of Victoria

Stack Exchange User Growth Analysis

Fall term 2023

CSC 501 Project 1 Report

Joshua Chapman
Reia Drucker
Paulina Gonzalez
Jaskaran Singh Purewal

December 6, 2023

1 Introduction

Stack Exchange is an internet forum website that allows users to post and answer questions about a variety of topics. The website releases data every three months that documents users, posts, tags, flags, votes, and a variety of other metrics [1]. This report details an analysis of the database providing novel insights into the data. Section 2 contains a triangulation of investigations centered around the change of individual users. Section 3 outlines the conceptual schema of the data used in the report. Section 4 proves the normalization qualities of the data set. Finally, section 5 theorizes the benefits and implementation of a graph based data model.

2 Data Analysis & Insights

Stack Exchange relies on users of the platform for content through questions and answers. We hypothesize that users improve as community members as they use the site for longer. Our investigation will test this hypothesis via a triangulation of four experiments. First, we evaluate the quality of posts over time. Second, the improvement in response time as users post more is measured. Third, exploration of whether there are any trends in the topics users post about, over time, is performed. Finally, the connection between time on the site and the awards users receive is evaluated.

2.1 Post Quality Over Time

This analysis aims to explore the quality of posts uploaded, both questions and answers, over time. The quality of a particular post is determined using the metric "Score". This score is the difference between the number of upvotes and the number of downvotes. We theorize that as the number of posts uploaded by a user increases, the corresponding score of the post improves.

Cleaning and evaluation of data was done using the following methods. Firstly the posts are categorized into questions and answers. Then the posts in each of these categories are grouped by their corresponding user, and each post for a particular user is appended in a list in the ascending order of the specific post creation. Finally, average score is calculated for all user's first, second, third and so on, posts. More simply, the first data point on the graph shows the average score of the first post of all users, and so on.

For the posts categorized as 'Questions':

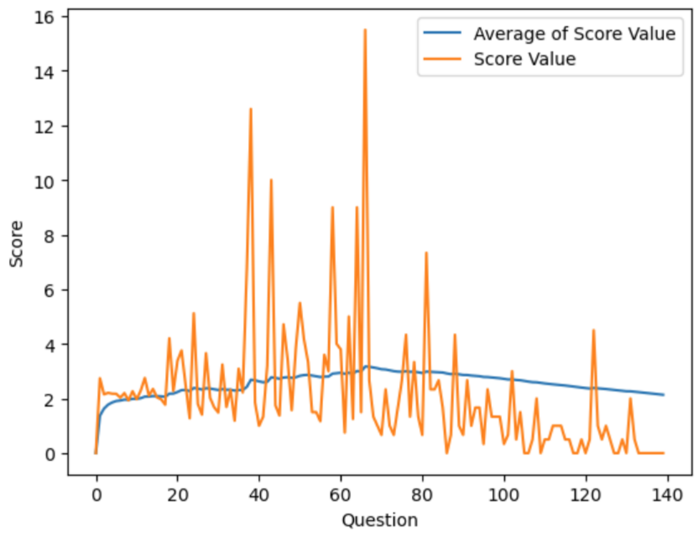

Figure 1: Evolution of Scores of Questions of all Users over time

The figure above depicts that as a user asks more questions over a period of time, the scores of the questions asked increase. This infers a positive trend until the 65th question asked, which supports our hypothesis. We see a decrease in average score after the 65th question because there are less than 3 users who have asked a total greater than 65 questions on the platform. Therefore, the majority of the data supports our hypothesis.

For the posts categorized as 'Answers':

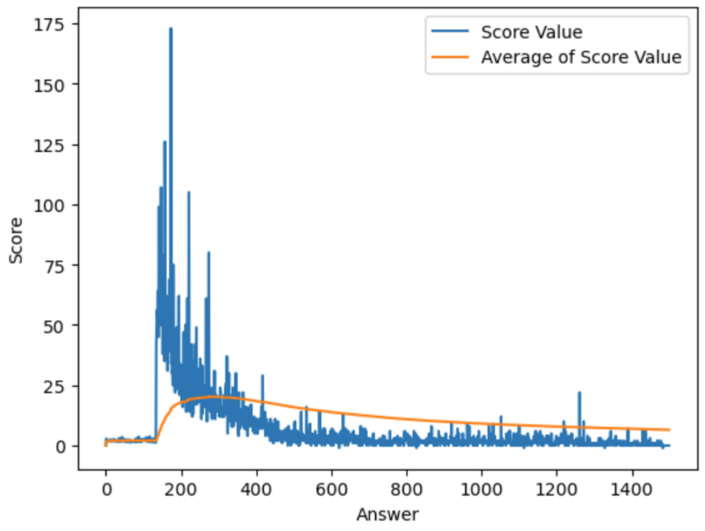

Figure 2: Evolution of Scores of Answers of all Users over time

The figure above depicts that as a user answers more questions over a period of time, the scores of the answer posts increase. This infers a positive trend until the 350th answer post, which supports our hypothesis. We see a decrease in average score after the 350th answer post because there are less than 2 users who have posted an answer for a total greater than 350 questions on the platform. There is also an unusual spike at the 150th answer mark. This unusual spike can be explained to the fact that there are a great number of users who post random answers or comments. This attributed noise negates the increase in average due to good scoring answers. But this noise stops at about the 150th answer mark. This means that the users who do not have interest in the topic tend to post at most 150 bad answers before stopping. This gives us a rough upper bound on the noisy data. It can also be inferred that only active and well-performing users post more than 150 answers on the portal. Therefore, the majority of the useful data supports our hypothesis.

The observed trend for both the 'Question' and 'Answer' posts shows an increase in the average score, as a user posts more over time. This signifies that the quality of questions posted as well as the answers posted increased over time, as a user posts more. We can infer an increase in the overall growth of a user in framing easy-to-understand correct responses that complement the user-base's knowledge regarding a particular topic of interest. Hence, it supports the hypothesis for this analysis.

2.2 Response Times to Users' Questions Over Time

This analysis aims to understand how the response time to questions evolves for active users on Stack Exchange. We hypothesized that as users become more established on the platform, the community's response time to their questions might change, possibly due to factors like growing user reputation or changing question complexity.

We filtered active users as those who asked more than 20 questions. For each user, we calculated the time to the first answer for all their questions. We employed different data visualization techniques, including line plots, bar charts, and rolling averages, to see the trends over time. Here are some of our findings:

- *Initial Response Time Analysis:* We first examined the response times for each user's questions, plotted against the date the questions were asked. This approach provided a good view of how quickly each question was answered, though the amount of data gives a complex picture.

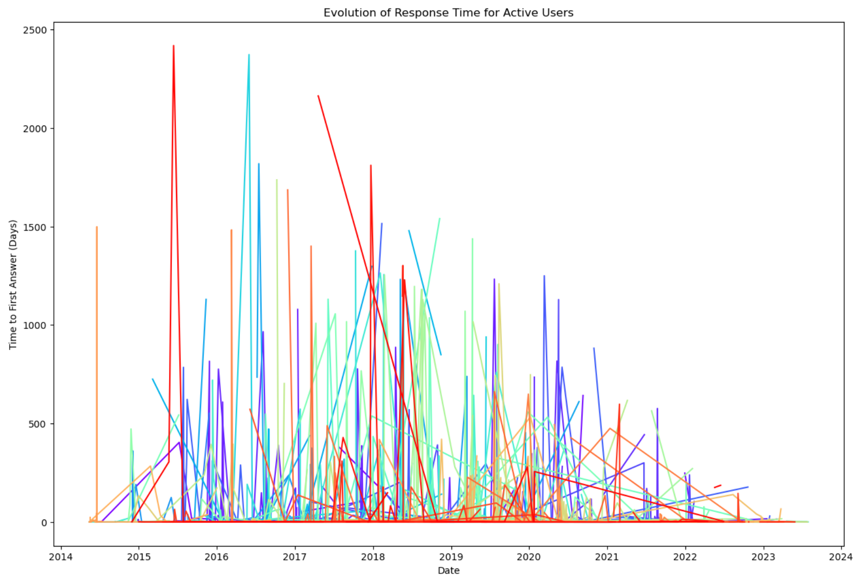

Figure 3: Evolution of Response Time for Active Users

- *Linear Regression Analysis:* A linear regression model was applied to have a better view of the general trend in response time. The model indicated a slight but statistically significant decrease in average response time over the user's lifespan on the platform.

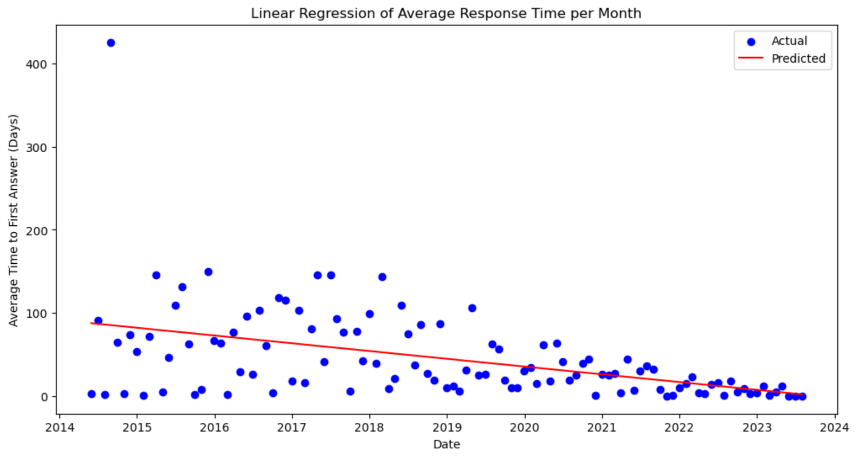

Figure 4: Linear Regression of Average Response Time

Our analysis reveals that active users on Stack Exchange tend to experience a change in the community's response time to their questions. Users initially facing longer response times saw a gradual decrease, reflecting their growing familiarity with the platform or the community's quicker engagement with their queries.

This trend may also be attributed to several key factors that indicate the users' development over time. Firstly, the improvement in response time could be linked to users formulating better-structured and more concise questions as they become better at identifying and articulating their issues. This clarity in question formulation likely makes it easier for other community members to provide relevant answers.

Moreover, the observed trend could suggest an overall enhancement in users' skills and understanding as data scientists or programmers. As users' expertise grows, their questions may become more focused and technically proficient, attracting quicker responses from knowledgeable community members who are more likely to engage with well-defined and challenging problems.

The decreasing response time may not only be a reflection of users becoming more integrated into the Stack Exchange community but also an indication of their personal growth and improvement in their respective fields. As users evolve from novices seeking basic guidance to more experienced members posing advanced queries, the community's response pattern adjusts accordingly, demonstrating a dynamic interaction between user expertise and community engagement.

2.3 User Topics Over Time

Given that Stack Exchange is often used by individuals as a tool to receive help from a community on the things they want to learn, we wanted to look into if the topics a user is interested change as they use the website more. For example, we were interested to see not just what the most popular topics of questions were, but to see the most popular progression of topics.

In order to do this we looked at the 10 most popular tags found on questions from valid users that had tags. Below is the list of the 10 most popular tags and how often they occur out of the 110,967 unique tags. Overall, these 10 tags are found in 37.6% of all questions.

Rank	**Tag Name**	**Count**
1	machine-learning	11,165
2	python	6,558
3	deep-learning	4,794
4	neural-network	4,319
5	classification	3,206
6	keras	2,715
7	nlp	2,640
8	scikit-learn	2,274
9	tensorflow	2,189
10	time-series	1,856

Table 1: The 10 most popular tags and the amount of times they occur in the data set.

After we found the 10 most popular tags, we grouped questions by the user who asked them and looked at counting the number of times a user's question's tags matched a pattern. The patterns we looked at were all possible orderings of length 4 of the 10 most popular tags, for a total of 10^4 different patterns. The most common patterns starting with each of the 10 most popular tags and their occurrence percentage among all users can be seen in the list below.

Pattern	**Occurrence Percentage**
machine-learning → machine-learning → machine-learning → machine-learning	0.654%
python → python → python → python	0.322%
deep-learning → deep-learning → deep-learning → deep-learning	0.19%
neural-network → machine-learning → machine-learning → machine-learning	0.146%
classification → machine-learning → machine-learning → machine-learning	0.122%
keras → keras → keras → keras	0.078%
nlp → nlp → nlp → nlp	0.078%
scikit-learn → python → python → python	0.078%
tensorflow → deep-learning → deep-learning → deep-learning	0.049%
time-series → time-series → time-series → time-series	0.039%

Table 2: Most common tag patterns of length 4 stemming from a first post where each of the top 10 tags was used.

The data here provides a few keys insights, the first being that users who post questions about popular topics tend to be relatively focused within that topic. Users with at least 4 posts, whose first post contained the tags *machine-learning, python, deep-learning, keras, nlp,* and *time-series* tended to stay within that topic. Exceptions are the *neural-network* and *classification* tags which switch quickly to *machine learning*, likely due to both tags being used in combination with the *machine-learning* tag. Classification and neural-networks are often used in introductory material to machine learning, so that pattern being popular makes sense as well. Other exceptions include the *scikit-learn* tag which switches to *python*, and *tensorflow* which switches to *deep-learning.*

What we found most interesting, was the *scikit-learn* → *python* path. The *scikit-learn* → *python* path is likely due to people who are just getting into machine learning with little experience programming in python, as people learning more specialized topics tend to have gaps in their general knowledge that they need to fill. The *tensorflow* → *deep-learning* path is a bit simpler and likely explained by the fact that tensorflow is often used as a library to tackle deep-learning problems.

Overall, the data suggests that users who ask more specific questions to start, tend to have some gaps in their general knowledge that they realize they need to fill, and users who ask more general questions don't tend to specialize that quickly. Further analysis could look at tag coexistence among the popular tags, with the combination of that data, along with this path data being useful for suggesting questions to certain users that they would likely be more interested in.

2.4 Badges Earned Over Time

This analysis aims to explore the type and number of badges earned by a user over time. There are three different types of badges that a particular user can achieve. Namely Gold, Silver and Bronze, in decreasing order of their difficulty to achieve. As described in [2], gold badges are awarded for the most difficult feats to the most committed users, silver badges are awarded to encourage more participation and are quite rare, and bronze badges are awarded for the basic use of the portal and are relatively easy for a user to receive.

Analysis Findings:

Badge	**Average Total Badges**
Gold	16.185
Silver	9.777
Bronze	2.228

Table 3: Average of total number of badges for a user having at least 1 given Badge

As the values in Table 3 suggest, a user with at least one gold badge, tends to have an average of 16 total badges, while a user with at least one silver badge tends to have a lower number, at an average of 9.7 total badges. If a user only has a bronze badge, that user tends to have only 2 total badges. This finding shows that as a user strives to achieve a more difficult badge, the user must be able to teach new users how the portal works, as well as take part in activities that are positive towards the community. Only then can a user gain enough easier to achieve badges and thus in due time, earn a gold badge.

Badge	**Average Number of Days to Achieve Badge**
Gold	1000.7623
Silver	733.7997
Bronze	78.4391

Table 4: Average number of days to achieve a badge

As the values in Table 4 suggest, it is evidently more time consuming to achieve gold and silver badges. This is directly related to the fact that if a user puts in more time and effort to learn particular skills, improve and build on their previous knowledge, and grow as a whole, in a particular field of interest, only then is that user awarded a gold or a silver badge.

Hence, the results from this analysis show that when a user puts in more time and effort on the portal to improve their knowledge, their growth is well reflected in number and type of badges that they receives.

3 Conceptual Design/Relational Model

The data analytics we wished to perform triangulated around the theme of users and their posts. This required data about posts and the users involved with those posts. The xml files provided by Stack Exchange [1], contained the data we needed for these analytics. They contained a variety of identifiers and metrics about the posts contained on the site and the associated users. We used a selection of attributes from these files for a variety of purposes and the ones we used in our analyses are depicted in our entity relationship diagram below.

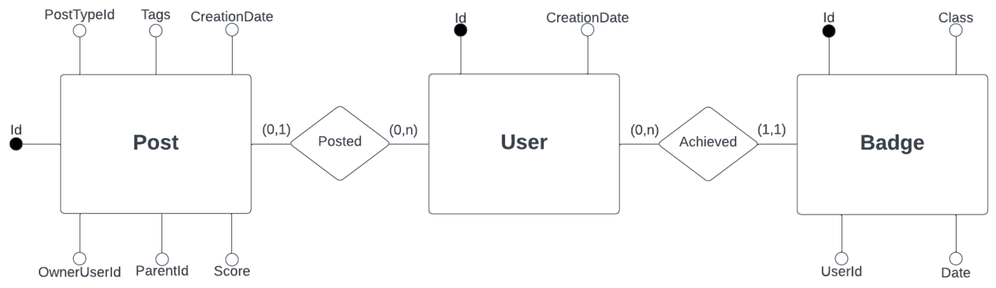

Figure 5: Entity Relationship Diagram

The entity relationship diagram clearly and effectively displays all of the information for the conceptual design. Additional details are provided below.

3.1 Entities

3.1.1 Post Attributes

- **Id (int)** is the identifier of the Posts entity. There is a unique identifier for each post in the database.
- **PostTypeId (tinyint)** is a classifier for the type of post. Ex. Questions = 1, Answers = 2, etc... [1] We used this to evaluate the growth of users dependent on the type of post.
- **Tags (list(string)** are classifiers of the subject matter of the post. There are thousands of tags, some examples are programming languages, topics or academic subjects. Tags were used in Investigation #3 where tag patterns for a single user were identified.
- **CreationDate (pandas.datetime)** contains the date and time of the post. All investigations used the creation date to chronologically sort posts.
- **OwnerUserId (int)** is a unique identifier from the Users table and allows us to associate posts with a specific user. This is used in the analysis to track posts associated with a specific User ID.
- **ParentId (int)** uses the format of UserID, similar to OwnerUserId. ParentId points to User ID of the original poster when the PostTypeId is an answer.
- **Score (int)** is a metric of posts. On stack exchange, users can give posts an "upvote" if they approve of the post. A post's score is the total number of these "upvotes". Score was one metric used to evaluate quality of a post.

3.1.2 User Attributes

- **Id (int)** is the identifier of the Users entity. There is a unique identifier for each User on the site.

- **CreationDate (pandas.datetime)** is the time the user was created on the site. These dates were used to measure the experience level of users.

3.1.3 Badge Attributes

- **Id (int)** is the identifier of the Users entity. A unique identifier is given for each Badge given to a user at a timestamp.
- **Class (tinyint)** is the quality of the badge. Gold = 1, silver = 2 and bronze = 3. Class was used in the investigations to evaluate the quality of users.
- **UserId (int)** is the ID of the user that achieved this badge. This ID allowed identification of badges to a user for investigation #4.
- **Date (pandas.datetime)** is the timestamp given at the time the badge was awarded.

3.2 Relationships and Cardinality

3.2.1 User Posted Posts

The Posted relationship outlines the action of a user making a post on Stack Exchange. It is a many-one or zero relationship. Users are able to have as many posts as they choose from 0 to n. Posts can have a maximum of one user associated with it. As well, posts are able to have no users attached to them. This occurs when a guest user makes a post.

3.2.2 User Achieved Badges

Users are able to achieve badges as they accomplish goals on Stack Exchange. A single user is able to possess multiple gold, silver or bronze awards. However, if a badge exists, then a user must be attached to it.

4 Normalisation

The schemas we used were:

$$\text{Posts(Id, PostTypeId, CreationDate, Score, OwnerUserId, Tags, ParentId)}$$

$$\text{Users(Id, CreationDate)}$$

$$\text{Badges(Id, Class, Date, UserId)}$$

To determine if a relation is in Boyce-Codd Normal Form (BCNF), we need to check for certain conditions. One of the key conditions for BCNF is that for every non-trivial functional dependency ($A \rightarrow B$), A must be a superkey. In our case, the non-trivial functional dependencies are:
For the Posts relation:

$$\{\text{Id}\} \rightarrow \{\text{PostTypeId, CreationDate, Score, OwnerUserId, Tags, ParentId}\}$$

For the Badges relation:

$$\{\text{Id}\} \rightarrow \{\text{Class, Date, UserId}\}$$

Important to note is that {CreationDate, OwnerUserId} $\rightarrow$ {Id, PostTypeId, Score, Tags, ParentId} is not a valid functional dependency because OwnerUserId could be null due to a guest user or a deleted user [3]. It should be evident that no other attributes or non-trivial combinations of attributes form any other functional dependencies.

The key for both relations is {Id} and since all keys are also superkeys, both relations are in BCNF because the determinant of the only functional dependency is a key. The same applies for the Users relation where {Id} is also a key and is trivialy in BCNF due to the only relation being {Id} $\rightarrow$ {CreationDate}.

5 Graph Data Modelling

The following is our proposed graph-based schema:

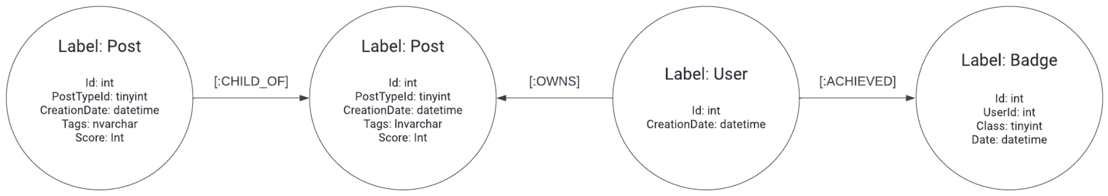

Figure 6: Graph Relationship Diagram

For the first analysis we conducted, the graph-based schema would better to support the analysis in the following ways. The graph-based model reduces the time complexities of certain operations such as filtering. This operation goes from being linearly dependent on the number of posts($O(n)$), to being dependent on the number of posts uploaded per user. This direct node-edge-node traversal in a graphical-based schema leverages connections between nodes to increase the speed of performing a query on a large database. Thus maximising performance in the case of real-world data.

For the second analysis we conducted, the graph-based schema would better support the analysis in the following ways. The graph-based model can more intuitively and directly represent the relationships between users and their questions. Each user and question can be a node, with edges representing the posting of a question. This direct mapping simplifies the process of tracking the journey of individual users, from their initial questions to their most recent ones, and analyzing the corresponding response times. In our analysis, we focus on how response times to a user's questions change over time. In a graph model, traversing from a user node to their sequential questions over time is more straightforward and efficient than executing time-based queries in a relational database.

For the relational model, operations like filtering questions and calculating response times are typically linear, with time complexities such as $O(n)$ for scanning the entire Posts table or $O(q)$ and $O(a)$ for filtering questions and answers respectively, where n represents the total number of posts, q the number of questions, and a the number of answers. However, this simplicity can lead to inefficiency as the size of the data-set grows. In contrast, the graph-based model often shows more favorable time complexities for similar operations. It leverages direct node-to-edge traversals, resulting in complexities like $O(k)$ for accessing a user's questions, where k is the average number of questions per user. This model maintains efficient querying performance even as data volume increases, making it particularly advantageous for large, interconnected data-sets.

For the third analysis we conducted, the graph-based schema would be better to support the analysis since the analysis is inherently driven by looking at the progression of a user's questions. In the relational model, in order to aggregate a user's questions we need to search the entire Posts table for all posts belonging to a user. This has a $O(|u| \cdot |p|)$ time complexity. In the graph-based model it is much simpler to aggregate a user's questions by following all OWNS edges of a user and checking the PostTypeId. The graph based model ensures that we only need to visit all of the posts once and therefore has time complexity $O(|p|)$. This is much more efficient than the relational model and scales much better, given that the underlying distribution of posts and users is a power law distribution where most users have few posts.

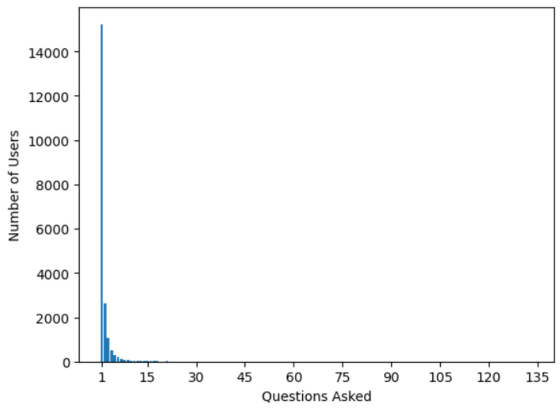

Figure 7: Graph demonstrating that as the number of questions asked increases, the number of users who have asked that many questions decreases following a power law distribution.

Similarly, for the fourth analysis we conducted, the graph-based schema would better to support the analysis in the following ways. As the data is naturally stored as hierarchical in the graph-based schema, with the data being highly connected, the retrieval of data by performing filtering queries is highly optimized as described for the other analysis as well. As the query is made across multiple tables, the performing multiple joins is slow and resource-intensive. Furthermore, the complexity of these operations increases exponentially as the data size increases. As the relationship between the users and the badges earned is stored, we get a much simpler graph model. The advantages of a graph data model are further realized when the data model requirements change. The graphical data-model, in which the attributes are not strictly defined, provides better flexibility and extensibility.

6 Conclusion

The stack exchange data provided unique insights into user development on the platform. Investigation into the score of posts over time measured a positive correlation between the scores and users posting more. This clearly demonstrates the connection between community recognition and valuation of frequent. Investigation #2 measured the correlation between a lower response time to posts as users posted more. This supports the idea that as users post more, they grow and have better interactions with the community. Investigation #3 evaluated patterns of the topics users posted about, and future topics they engaged with. Users tended to follow search patterns that indicated development and a search for a deeper understanding of the topics. Lastly, the investigation of badges earned over time clearly demonstrated a growth of users. The badges metric was a clear indicator that growth is a result of time spent on the platform. Overall, our investigations support the conclusion that users that spend more time on the platform become more experienced and are rewarded by the built in metrics of the platform.

References

[1] leerssej, "Database schema documentation for the public data dump and sede," Aug 1955.

[2] "What are badges? - help center — math.stackexchange.com." [Accessed 06-12-2023].

[3] A. Badia and D. Lemire, "Functional dependencies with null markers," *The Computer Journal*, vol. 58, p. 1160–1168, May 2014.