Synthetische Datengenerierung aus PostgreSQL für PostgreSQL

OST Ostschweizer Fachhochschule

Studienarbeit Herbstsemester 2021

Themengebiet Software Engineering / Data Engineering

Jari Elmer

Timon Erhart

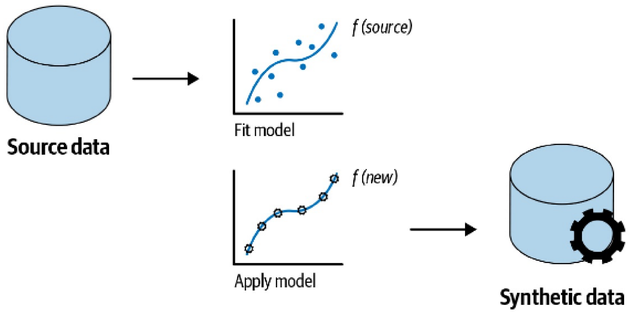
Betreuer: Nicola Jordan

Experte: Prof. Stefan Keller

Projektpartner: Institut für Software

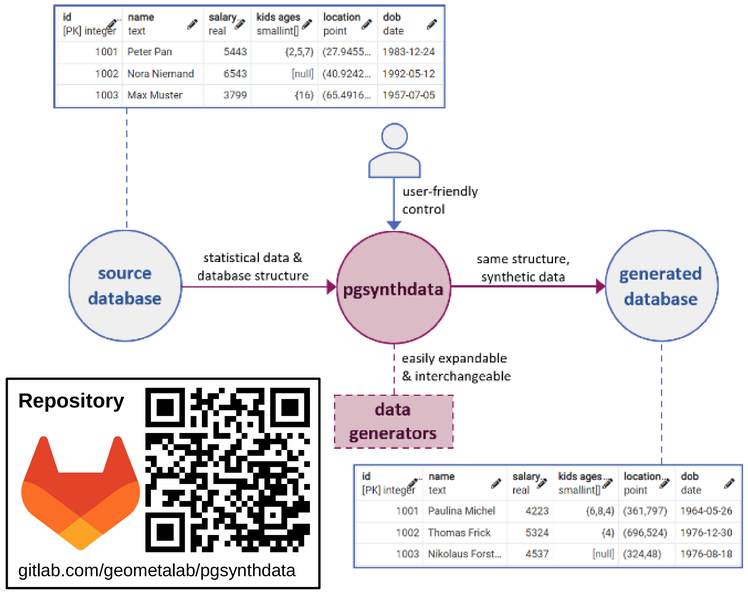
In den Bereichen Software- und Data-Engineering sowie Machine-Learning besteht eine grosse Nachfrage nach umfangreichen Datensätzen. Dabei ist der Datenschutz oftmals ein Grund, keine Realdaten zu verwenden. Eine Pseudonymisierung bietet keinen vollständigen Schutza vor De-Anonymisierungs-Attacken. Eine komplette Anonymisierung wäre eine mögliche Lösung. Doch diese ist aufwändig und zeitintensiv.

Am vielversprechendsten sind rein synthetisch generierte Daten mit ähnlichen statistischen Eigenschaften wie die Originaldaten. Das hat den zusätzlichen Vorteil, dass beliebige Datenmengen erzeugt werden können.



From *Practical Synthetic Data Generation*, by Khaled El Emam, Lucy Mosquera, and Richard Hoptroff. Copyright © 2020 K Sharp Technology Inc., Lucy Mosquera, and Richard Hoptroff. Published by O'Reilly Media, Inc. Used with permission

Ein an der OST entwickelter Softwareprototyp pgsynthdata setzt darum auf das Prinzip der rein synthetischen Datengeneratoren. Das Kommandozeilen-Tool ist in der Lage, fast beliebige PostgreSQL-Datenbanken zu synthetisieren und in eine generierte Datenbank mit gleicher Struktur und vergleichbaren statistischen Eigenschaften abzufüllen. Als Grundlage dient der Katalog von PostgreSQL für funktionale Abhängigkeiten sowie Statistiken, die PostgreSQL zum Zwecke der Anfrageoptimierung erstellt. Ein Alleinstellungsmerkmal des Tools ist, dass es ohne aufwändige manuelle Konfiguration auskommt.

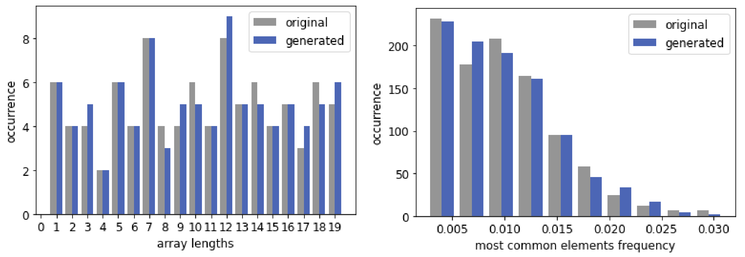


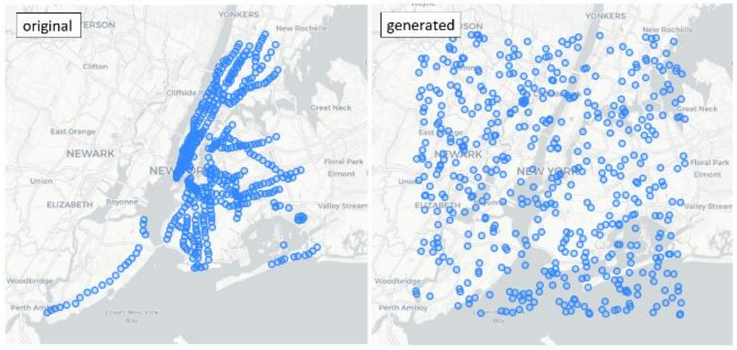
Das Ziel dieser Arbeit war es, die bestehende Software pgsynthdata in einen wartbaren, erweiterbaren und einfach zu nutzenden Zustand zu überführen. Zudem wurde die Funktionalität um Generatoren für Datentypen wie enumerated Types (Enum), Arrays und Geometrien erweitert.

Vorgehen und Resultate

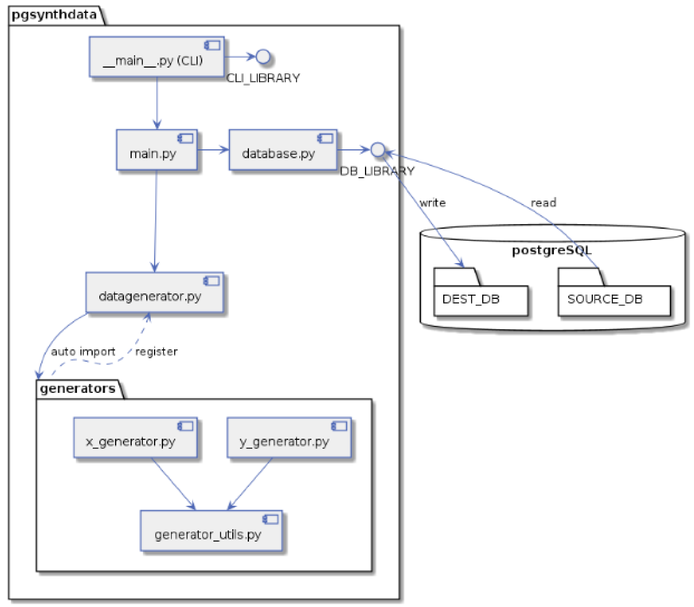
Um das bestehende Tool in den durch die Aufgabenstellung definierten Zustand zu bringen, benötigte es zu Beginn ein umfassendes Refactoring des gesamten Tools. Der Fokus lag dabei auf einem Plugin-System, welches es ermöglicht, neue Generatoren einfacher zu integrieren. Um eine saubere Architektur mit klaren Verantwortlichkeiten zu erhalten, wurde der komplette Programmablauf neu geschrieben. Durch diese Entkopplung der Module konnte auch die Testabdeckung stark verbessert werden.

Nach der Übernahme der bestehenden Generatoren ins neue System, wurde ein Array-Generator für numerische Typen, ein Generator für PostGIS-Geometrie-Typen und ein Enum-Generator implementiert. Dies war in zweierlei Hinsicht wichtig: Zum einen wurde das Tool weiterentwickelt und unterstützt nun mehr Datentypen, zum anderen konnten die Funktionalität und Flexibilität des Plugin-Systems überprüft werden.



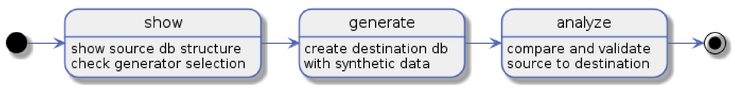


Aus dem Bedürfnis heraus, statt zufälliger Zeichenketten realistische Eigennamen generieren zu können, ist eine Möglichkeit entstanden, per Spalten-Kommentar im Schema spezifische Generatoren zu konfigurieren. Die implementierten Generatoren für Eigennamen und Postadressen verdeutlichen dieses Konzept. Das Konfigurieren per Kommentar hat sich als benutzerfreundlich und unkompliziert bewährt, ausserdem benötigt es keine speziellen Fachkenntnisse.



Fazit und Ausblick

Entstanden ist ein wartbares und erweiterbares Programm mit einem flexiblen Plugin-System und mit Generatoren für eine Vielzahl von Datentypen. Constraints wie Unique- oder Foreign-Keys werden unterstützt. Bei nicht unterstützten Datentypen wird dem User eine entsprechende Warnung angezeigt. Die neuen Kommandos show und analyze ermöglichen einen Workflow für die Erzeugung synthetischer Daten mit anschliessender Analyse.



Das Tool ist im Stande, aus verschiedenen PostgreSQL-Datenbanken synthetische Daten zu generieren. Weiterhin gibt es viele denkbare Erweiterungen, um die Funktionalität des Tools zu erhöhen sowie mehr Datentypen zu unterstützen. Als Beispiel wäre eine Unterstützung von zusammengesetzten Primärschlüssel denkbar.

Zudem hat es sich gezeigt, dass die konfigurierbare Auswahl der Generatoren interessante neue Möglichkeiten eröffnet. Dieses Konzept könnte noch weiterverfolgt werden. So wäre es denkbar, dass spezifische Generatoren nicht nur über die Konfiguration ausgewählt, sondern auch gleich noch parametrisiert werden. Zum Beispiel wäre eine sprachregionsspezifische Parametrisierung für Personen- und Strassennamen interessant.