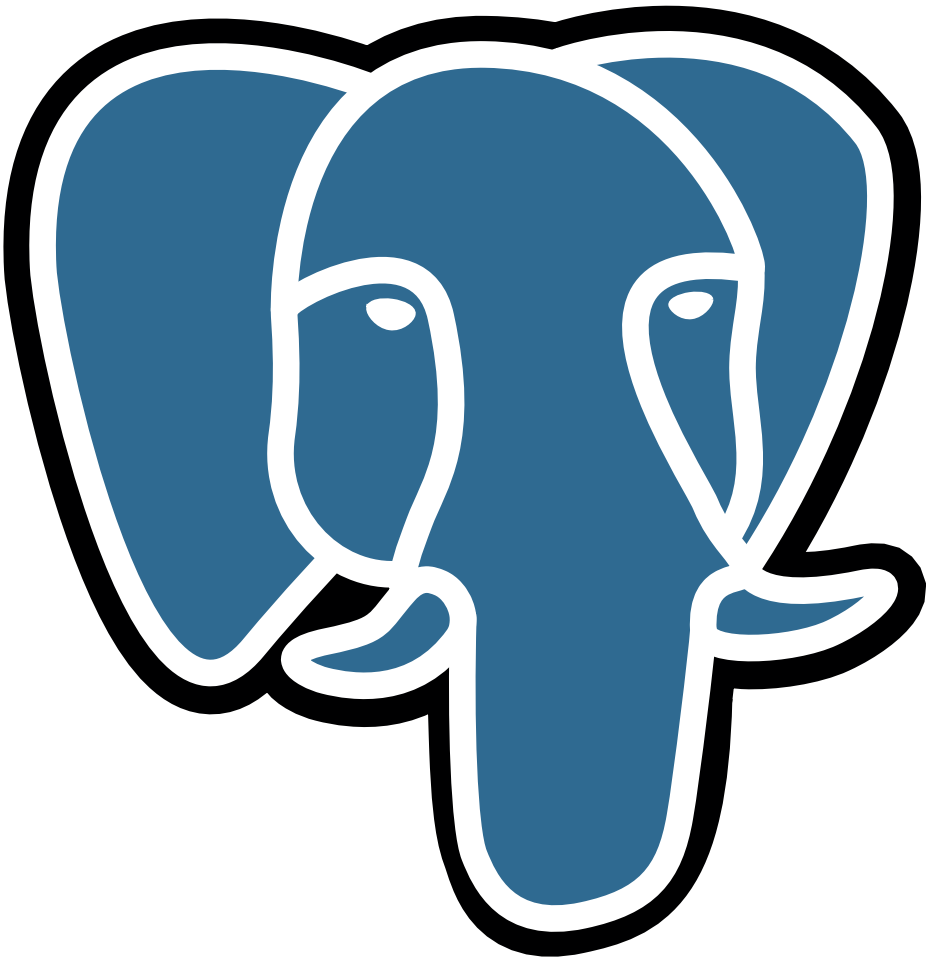

PostgreSQL – der neue Standard für Allzweck-Datenbanken

Jan Karremans, EnterpriseDB

Jeder hat den Namen wahrscheinlich schon mal irgendwo gehört: PostgreSQL oder kurz: Postgres. Aber was ist das eigentlich? Das hat doch irgendetwas mit Open Source zu tun? Bei einer Datenbank erscheint das zunächst ein bisschen merkwürdig. So etwas Hochkompliziertes wie eine hochverfügbare, geschäftskritische Fünf-Terabyte-Datenbank, gebaut von irgendwelchen dubiosen Hoodie-Trägern in schummerigen Kellern, darauf soll sich ein Unternehmen verlassen?

Mitte der siebziger Jahre arbeiteten die Herren Edgar Codd und Christopher Date bei IBM an der Umsetzung des von Codd im Jahr 1970 vorgestellten relationalen Datenmodells. Innerhalb dieses Forschungsprojekts entstand die erste tatsächliche Implementierung von SQL. Wikipedia meldet dazu *[1]*: „System R ist ein von IBM im San Jose Research Center (heute IBM Almaden Research Center) entwickeltes Datenbank-Managementsystem. Das System wurde in den 1970er Jahren im Rahmen eines Forschungsprojekts entwickelt. Es ist historisch von großer Bedeutung, da es das erste relationale Datenbank-Managementsystem war und die Abfragesprache SEQUEL (= Structured English Query Language) definierte, aus der die SQL-Abfragesprache hervorging. Durch die Erfahrungen mit dem System R wurde von der IBM das System SQL/DS entwickelt und ab dem Jahr 1981 bei Kunden eingesetzt. SQL/DS ist der Vorgänger des relationalen Datenbank-Managementsystems DB2."

Es ist interessant zu sehen, dass die relationale Theorie erfunden wurde, um die Probleme mit NoSQL-Datenbanken, die es damals nämlich schon gab, zu beheben. Basierend auf dieser Theorie wurden unterschiedliche Projekte gestartet, von denen Relational Software, im Jahr 1977 gegründet – jetzt besser bekannt als „Oracle" –, das erste richtige kommerziell erfolgreiche Produkt war.

Im Jahr 1973 veröffentlichte IBM das erste Research Paper über das System-R-Projekt. Michael Stonebraker und Eugene Wong von der Berkeley University waren daran stark interessiert und haben darauf basierend das Projekt „University Ingres" oder „Berkeley Ingres" begonnen. Nach der Fertigstellung des ersten Prototyps im Jahr 1974 wurde das Ingres-Projekt noch bis zum Jahr 1985 an der Uni weitergeführt. Michael Stonebraker startete im Jahr 1985 das Post-Ingres-Projekt mit dem Ziel, die Probleme von derzeitigen Datenbank-Systemen zu beseitigen.

Der erste Prototyp erschien im Jahr 1987. Auf Version 1 im Juni 1989 folgten genau ein Jahr später Version 2 und im Jahr 1991 Version 3. Das Universitätsprojekt endete am 30. Juni 1994. Die Jahre 1994 bis 1996 wurden genutzt, um SQL als Abfragesprache zu implementieren. Die Postgres-Community nahm am 8. Juli 1996 den ersten Entwicklungsserver außerhalb einer Universität in Betrieb.

Postgres ist nicht nur ein Open-Source-Projekt, sondern sogar ein Community-Open-Source-Projekt. Die Entwicklung von Postgres erfolgt völlig offen und unabhängig, anders als etwa MySQL oder MongoDB, deren Weiterentwicklung von Unternehmen getrieben und gesteuert wird *[2]*. Seit dem Jahr 1996 steuert ein „PostgreSQL Core Team" das Projekt. „The PostgreSQL Global Development Group" erhält tatkräftige Unterstützung von Major- und Regular-Contributors *[3]*.

Open-Source-Wellen

Abbildung 1 sieht schön bunt aus. Aber was bedeutet das? Kommerzielle Anwen-

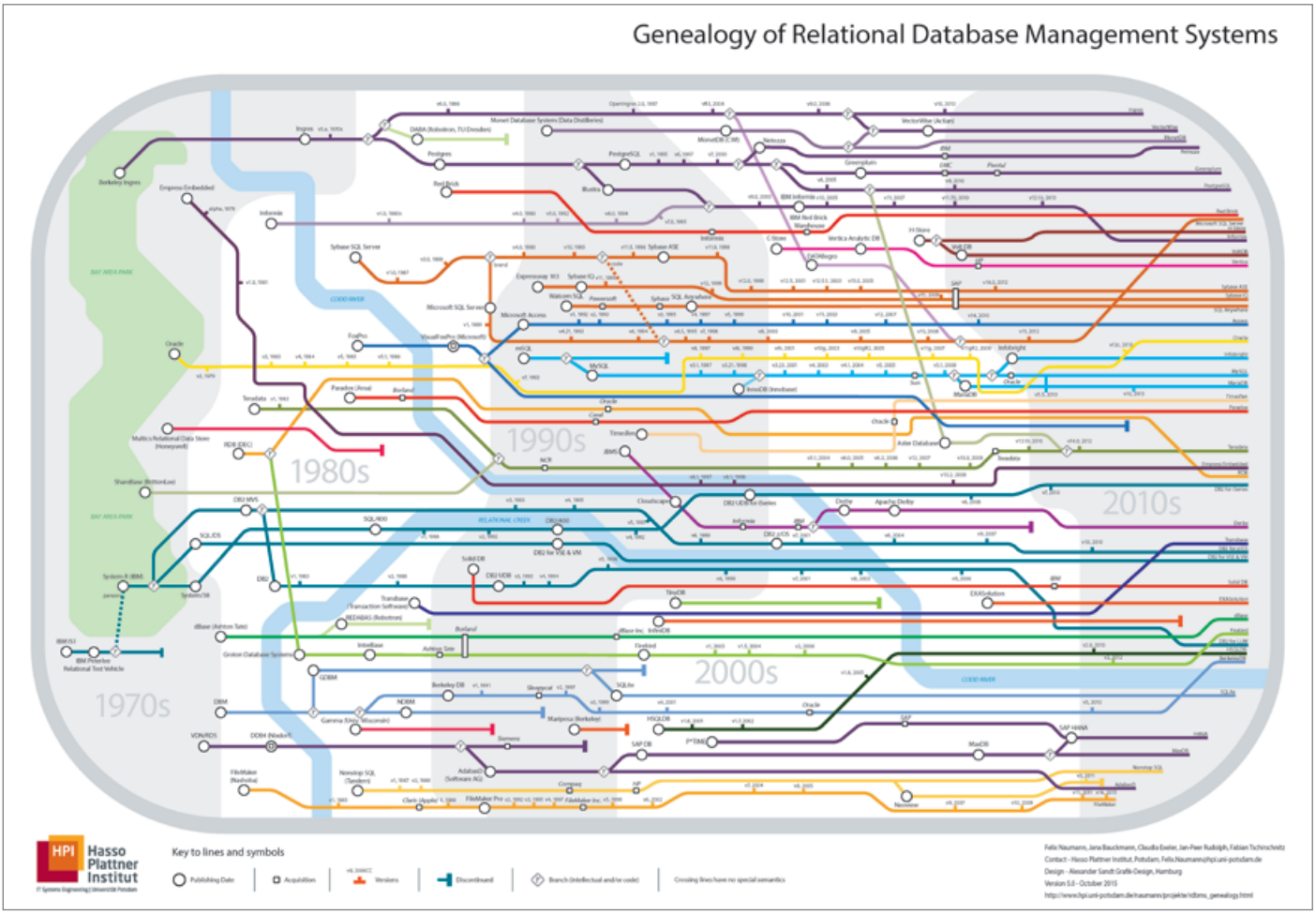

Abbildung 1: Stammbaum der relationalen Datenbank-Systeme

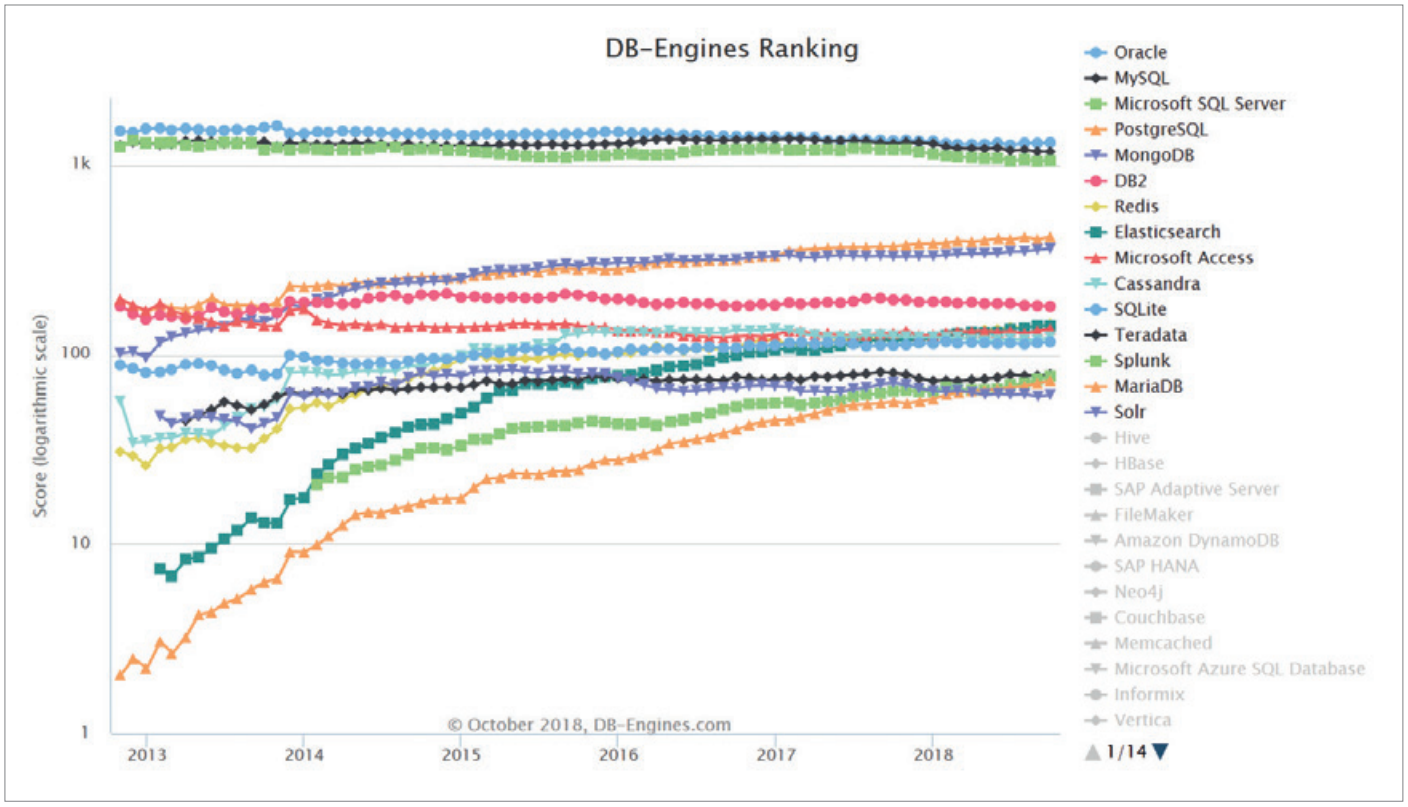

Abbildung 2: Die Verbreitung der Datenbanken

dungen sind seit vielen Jahren sehr erfolgreich. Sie werden auch in Zukunft erfolgreich sein, aber immer mehr aus dem Mainstream verdrängt. Die erste Open-Source-Welle hat gezeigt, worum es geht: Heute wird der größte Teil aller Server mit Linux, einem Open-Source-Betriebssystem, gefahren. Traditionelle Betriebssysteme, die Unternehmen bislang erfolgreich beim Wachstum unterstützt haben, wie die unterschiedlichen Varianten von Unix und Microsoft Windows Server, wurden aus dem Mainstream verdrängt und überleben in der Nische.

Das Gleiche passiert gerade ein weiteres Mal: Die zweite Open-Source-Welle rollt – Datenbanken werden mehr und mehr zum Mainstream. Dieselbe Bewegung, die zuvor Linux groß gemacht hat, bringt jetzt Open-Source-Datenbanken hervor *[4]*.

Der Blick auf die Geschichte hat gezeigt, dass RDBMS entwickelt wurden, um die Probleme der hierarchischen NoSQL-Datenbanken zu lösen. Letztendlich werden Open-Source-Anwendungen für Mainstream-Problembereiche immer mehr an Kraft und Einfluss gewinnen.

PostgreSQL wird unter der PostgreSQL-Lizenz veröffentlicht, einer liberalen Open-Source-Lizenz, ähnlich der BSD- oder MIT-Lizenz. Die Verbreitung von Postgres steigt insbesondere in Deutschland enorm. Postgres-Nutzer brauchen sich allerdings nirgends zu registrieren.

Deshalb ist leider unbekannt, wie viele Postgres-Instanzen es tatsächlich gibt. Ein schöner Indikator ist die Webseite „DB-Engines.com“. Sie ermittelt regelmäßig ein Ranking über die Verbreitung von Datenbanken *(siehe Abbildung 2)*.

Der von DB Engines verwendete Algorithmus basiert auf folgenden Faktoren:

- Anzahl von Erwähnungen auf Webseiten
- Generelles Interesse
- Anzahl der technischen Diskussionen
- Menge an Stellenangeboten
- Menge an Referenzen in Bewerbungen
- Relevanz in sozialen Netzwerken

Postgres wurde von DB Engines im Jahr 2017 zur Datenbank des Jahres gekürt. Daraus kann jeder seine eigenen Schlüsse ziehen.

Was hinter Postgres steckt

Postgres ist ein relationales Datenbank-Management-System (RDBMS). Es ist genau wie beispielsweise Oracle, Microsoft SQL Server oder IBM DB2 eine Multi-Purpose-Datenbank. Man kann also mit einer solchen Datenbank viele verschiedene Anwendungsszenarien abdecken. Dieser Datenbank-Typ ist am weitesten verbreitet.

Postgres verwaltet neben relationalen Daten auch „key-value pairs“. Es kann als Data Warehouse und Document Store eingesetzt werden und hat spezielle Datentypen und eine umfangreiche Unterstützung für geografische (Spatial-) Daten.

Postgres ist als erweiterbare Datenbank entworfen worden. Michael Stonebraker hat schon bei der Entwicklung von University Ingres festgelegt, dass seine relationale Datenbank erweiterbar sein muss. PostgreSQL ist so konzipiert, dass es leicht erweiterbar ist. Aus diesem Grund können in die Datenbank geladene, selbst geschriebene Erweiterungen genauso funktionieren wie eingebaute Features.

Extensions können in verschiedenen Programmiersprachen entwickelt werden. Solche Erweiterungen vergrößern die Funktionalität des Datenbank-Kerns. Da sich eine große Gruppe von Postgres-Entwicklern mit solchen Erweiterungen beschäftigt, sind unzählige Erweiterungen fix und fertig verfügbar.

```
select distinct product_type
     , data->>'brand'        as Brand
     , data->>'available'    as Availability
  from json_data
  join products
    on (
       products.product_type = json_data.data->>'name'
       )
 where json_data.data->>'available' = true
;
```

Listing 1

Postgres ist einfach. Die Postgres-Entwickler haben sich zum Ziel gesetzt, alle notwendige Komplexität in der Datenbank zu verstecken. Das führt dazu, dass Installation und Betrieb einer Postgres-Datenbank sehr einfach sind. So einfach, dass man vielleicht im Anfang denkt: „Das ist so einfach, das kann doch gar nichts sein." Trotzdem funktioniert eine Postgres-Datenbank mindestens genauso gut wie die direkte Konkurrenz.

Diese Einfachheit führt dazu, dass sich Postgres hervorragend in die aktuellen Trends im IT-Sektor einfügt: Die Postgres-Datenbank verhält sich außergewöhnlich gut in einem DevOps-Umfeld. Postgres scheint wie gemacht zu sein für den Einsatz in einer Container-Infrastruktur, da die Installationsgröße angenehm gering ist. Auch für den Einsatz in der Cloud ist Postgres sehr gut geeignet.

Postgres vermischt relationale und NoSQL-Daten. Die native Erweiterbarkeit von Postgres hat den Einbau von NoSQL-Funktionalität sehr einfach gemacht, als diese zu Beginn der 2000er Jahre wieder populär wurde. Mit dem Aufkommen von JSON wurden ab dem Jahr 2010 für Postgres die JSON- und JSONB-Dateitypen entwickelt. Die Erweiterbarkeit von Postgres hat es ermöglicht, dass zusätzlich Operatoren und Funktionen entstanden sind, um die Benutzung von JSON innerhalb der Datenbank zu vereinfachen. Es können Indizes auf JSON-Datentypen gebaut werden, die einzelne Felder indexieren. So lässt sich JSON mit relationalen Daten in einer einzigen Abfrage verknüpfen *(siehe Listing 1)*.

Postgres im Produktiv-Einsatz

Vor dem Einsatz von Postgres in „Mission Critical"-Umgebungen gibt es noch eine letzte Hürde. Eine „Community Open Source"-Lösung hat ihre Vorteile. Sie hat aber auch, so wie die meisten Open-Source-Lösungen, eine Schwachstelle: den Support; insbesondere dann, wenn es sich um einen 24/7-Support für geschäftskritische Systeme handeln soll.

Für den Einsatz von Open-Source-Lösungen im Unternehmen ist diese Hersteller-ähnliche Unterstützung von kritischer Wichtigkeit. Im Linux-Umfeld haben Unternehmen wie Red Hat oder Suse diese Rolle übernommen und bieten den passenden Support an. Im europäischen Raum bieten die Unternehmen „2nd Quadrant" und „EnterpriseDB" diese Unterstützung im Postgres-Umfeld. Die Firma „2nd Quadrant" ist auf den Support von Postgres spezialisiert; sie unterstützt und führt die Postgres-Community. „EnterpriseDB" ist zusätzlich auf Migrationen spezialisiert und begleitet Kunden, die Postgres als Ersatz für Legacy-Datenbank-Umgebungen einsetzen wollen, auf ihrem Weg von Oracle, MS SQL oder IBM zu Postgres. Dazu entwickelt EnterpriseDB die Oracle-Kompatibilitätsschicht, eine Erweiterung von Postgres.

Eine Kombination aus nativen und integrierten Tools ermöglicht den hochverfügbaren Betrieb von Postgres, ähnlich wie die Active-Data-Guard-Lösung von Oracle. Für Backup und Recovery ist ein skriptierbares Tool mit ähnlicher Funktionalität verfügbar, vergleichbar mit RMAN bei Oracle. Monitoring und Wartung der Postgres-Datenbank können komplett über Tools wie den Postgres Enterprise Manager erfolgen. So kann mit Fug und Recht behauptet werden: „Es gibt eigentlich keinen Grund mehr, Postgres nicht zu benutzen!"

Hinweis: Redaktionelle Bearbeitung von Sabine Heimsath und Robert Marz, DOAG Development Community

Referenzen

[1] *https://de.wikipedia.org/wiki/IBM_System_R*
[2] *https://en.wikipedia.org/wiki/PostgreSQL*
[3] *https://www.postgresql.org/community/contributors*
[4] *https://momjian.us/main/writings/pgsql/forever.pdf*

Jan Karremans
jan.karremans@enterprisedb.com