Menerapkan PostgreSQL terkelola untuk aplikasi SaaS multi-tenant AWS

AWS Bimbingan Preskriptif

AWS Bimbingan Preskriptif: Menerapkan PostgreSQL terkelola untuk aplikasi SaaS multi-tenant AWS

Table of Contents

Menerapkan PostgreSQL terkelola untuk aplikasi SaaS multi-tenant AWS

Tabby Ward dan Thomas Davis, Amazon Web Services ()AWS

April 2024 (riwayat dokumen)

Ketika Anda memilih database untuk menyimpan data operasional, penting untuk mempertimbangkan bagaimana data harus disusun, pertanyaan mana yang akan dijawabnya, seberapa cepat akan memberikan jawaban, dan ketahanan platform data itu sendiri. Selain pertimbangan umum ini adalah implikasi perangkat lunak sebagai layanan (SaaS) untuk data operasional, seperti isolasi kinerja, keamanan penyewa, dan karakteristik unik serta pola desain yang khas dari data untuk aplikasi SaaS multi-penyewa. Panduan ini membahas bagaimana faktor-faktor ini berlaku untuk menggunakan database PostgreSQL di Amazon Web AWS Services () sebagai penyimpanan data operasional utama untuk aplikasi SaaS multi-penyewa. Secara khusus, panduan ini berfokus pada dua opsi PostgreSQL AWS terkelola: Amazon Aurora PostgreSQL Compatible Edition dan Amazon Relational Database Service (Amazon RDS) untuk PostgreSQL.

Hasil bisnis yang ditargetkan

Panduan ini memberikan analisis terperinci tentang praktik terbaik untuk aplikasi SaaS multi-penyewa menggunakan Aurora PostgreSQL kompatibel dan Amazon RDS for PostgreSQL. Kami menyarankan Anda menggunakan pola dan konsep desain yang disediakan dalam panduan ini untuk menginformasikan dan membakukan implementasi Aurora PostgreSQL yang kompatibel dengan Aurora atau Amazon RDS for PostgreSQL untuk aplikasi SaaS multi-penyewa Anda.

Panduan preskriptif ini membantu mencapai hasil bisnis berikut:

- Memilih opsi PostgreSQL AWS terkelola yang paling optimal untuk kasus penggunaan Anda — Panduan ini membandingkan opsi relasional dan non-relasional untuk penggunaan database dengan aplikasi SaaS. Ini juga membahas kasus penggunaan mana yang paling optimal untuk Aurora PostgreSQL kompatibel dan Amazon RDS for PostgreSQL. Informasi ini akan membantu dalam memilih opsi terbaik untuk aplikasi SaaS Anda.
- Penegakan praktik terbaik SaaS melalui adopsi model partisi SaaS - Panduan ini membahas dan membandingkan tiga model partisi SaaS luas yang berlaku untuk sistem manajemen basis data PostgreSQL (DBMS): model kumpulan, jembatan, dan silo, dan variasinya. Pendekatan ini

menangkap praktik terbaik SaaS dan memberikan fleksibilitas saat merancang aplikasi SaaS. Penegakan model partisi SaaS adalah bagian penting dari melestarikan praktik terbaik.

- Penggunaan RLS yang efektif dalam model partisi SaaS kolam - Keamanan tingkat baris (RLS) mendukung penegakan isolasi data penyewa dalam satu tabel PostgreSQL dengan membatasi baris yang dapat dilihat berdasarkan pengguna atau variabel konteks. Saat Anda menggunakan model partisi kolam, RLS diperlukan untuk mencegah akses penyewa silang.

Memilih database untuk aplikasi SaaS

Untuk banyak aplikasi SaaS multi-tenant, memilih database operasional dapat disuling menjadi pilihan antara database relasional dan non-relasional, atau kombinasi keduanya. Untuk membuat keputusan, pertimbangkan persyaratan dan karakteristik data aplikasi tingkat tinggi ini:

- Model data aplikasi
- Pola akses untuk data
- Persyaratan latensi basis data
- Integritas data dan persyaratan integritas transaksional (atomisitas, konsistensi, isolasi, dan daya tahan, atau ACID)
- Ketersediaan Lintas Wilayah dan persyaratan pemulihan

Tabel berikut mencantumkan persyaratan dan karakteristik data aplikasi, dan mendiskusikannya dalam konteks penawaran AWS database: Aurora PostgreSQL kompatibel dan Amazon RDS for PostgreSQL (relasional), dan Amazon DynamoDB (non-relasional). Anda dapat mereferensikan matriks ini ketika Anda mencoba memutuskan antara penawaran basis data operasional relasional dan non-relasional.

Basis Data	Persyaratan dan karakteristik data aplikasi SaaS				
	Model data	Pola akses	Persyaratan latensi	Integritas data dan transaksional	Ketersediaan dan pemulihan Lintas Wilayah
Relasional (Aurora PostgreSQL kompatibel dan Amazon RDS untuk PostgreSQL)	Relasional atau sangat dinormalisasi.	Tidak harus direncanakan secara menyeluruh sebelumnya.	Lebih disukai toleransi latensi yang lebih tinggi; dapat mencapai latensi yang lebih rendah	Data tinggi dan integritas transaksional dipertahankan secara default.	Di Amazon RDS, Anda dapat membuat replika baca untuk penskalaan dan failover

			secara default dengan Aurora dan dengan menerapkan replika baca, caching, dan fitur serupa.		lintas wilayah. Aurora sebagian besar mengotomatiskan proses ini. Untuk konfigurasi aktif-aktif di beberapa Wilayah AWS, Anda dapat menggunakan penerusan tulis bersama dengan database global Aurora.
Non-relasional (Amazon DynamoDB)	Biasanya didenormalisasi. Database ini memanfaatkan pola untuk pemodelan many-to-many hubungan, item besar, dan data deret waktu.	Semua pola akses (query) untuk data harus dipahami secara menyeluruh sebelum model data diproduksi.	Latensi sangat rendah dengan opsi seperti Amazon DynamoDB Accelerator (DAX) mampu meningkatkan kinerja lebih jauh.	Integritas transaksional opsional dengan mengorbankan kinerja. Masalah integritas data dialihkan ke aplikasi.	Pemulihan lintas wilayah yang mudah dan konfigurasi aktif-aktif dengan tabel global. (Kepatuhan ACID hanya dapat dicapai dalam satu AWS Wilayah.)

Beberapa aplikasi SaaS multi-tenant mungkin memiliki model data unik atau keadaan khusus yang lebih baik dilayani oleh database yang tidak termasuk dalam tabel sebelumnya. Misalnya, kumpulan data deret waktu, kumpulan data yang sangat terhubung, atau memelihara buku besar transaksi terpusat mungkin memerlukan menggunakan jenis database yang berbeda. Menganalisis semua kemungkinan berada di luar cakupan panduan ini. Untuk daftar lengkap penawaran AWS database dan bagaimana mereka dapat memenuhi kasus penggunaan yang berbeda pada tingkat tinggi, lihat bagian Database dari whitepaper Ikhtisar Amazon Web Services.

Sisa dari panduan ini berfokus pada layanan database AWS relasional yang mendukung PostgreSQL: Amazon RDS dan Aurora PostgreSQL kompatibel. DynamoDB memerlukan pendekatan yang berbeda untuk mengoptimalkan aplikasi SaaS, yang berada di luar cakupan panduan ini. Untuk informasi selengkapnya tentang DynamoDB, lihat posting blog Mempartisi Data SaaS AWS Multi-Penyewa yang Dikumpulkan dengan Amazon DynamoDB.

Memilih antara Amazon RDS dan Aurora

Dalam kebanyakan kasus, sebaiknya gunakan Aurora PostgreSQL yang kompatibel dengan Amazon RDS for PostgreSQL. Tabel berikut menunjukkan faktor-faktor yang harus Anda pertimbangkan ketika memutuskan antara dua opsi ini.

Komponen DBMS	Amazon RDS untuk PostgreSQL	Kompatibel dengan Aurora PostgreSQL
Skalabilitas	Jeda replikasi menit, maksimal 5 replika baca	Kelambatan replikasi kurang dari satu menit (biasanya kurang dari 1 detik dengan database global), maksimum 15 replika baca
Pemulihan kerusakan	Pos pemeriksaan terpisah 5 menit (secara default), dapat memperlambat kinerja database	Pemulihan asinkron dengan thread paralel untuk pemulihan cepat
Kegagalan	60-120 detik selain waktu pemulihan crash	Biasanya sekitar 30 detik (termasuk pemulihan crash)

Komponen DBMS	Amazon RDS untuk PostgreSQL	Kompatibel dengan Aurora PostgreSQL
Penyimpanan	IOPS maksimum 256.000	IOPS dibatasi hanya oleh ukuran dan kapasitas instans Aurora
Ketersediaan tinggi dan pemulihan bencana	Dua Availability Zone dengan instance siaga, failover lintas wilayah untuk membaca replika atau salinan cadangan	Tiga Availability Zone secara default, failover lintas wilayah dengan database global Aurora, tulis penerusan untuk konfigurasi aktif-aktif Wilayah AWS
Cadangan	Selama jendela cadangan, dapat memengaruhi kinerja	Pencadangan inkremental otomatis, tidak ada dampak kinerja
Kelas instance database	Lihat daftar kelas instans Amazon RDS	Lihat daftar kelas instance Aurora

Dalam semua kategori yang dijelaskan dalam tabel sebelumnya, Aurora PostgreSQL kompatibel biasanya pilihan yang lebih baik. Namun, Amazon RDS for PostgreSQL mungkin masih masuk akal untuk beban kerja kecil hingga menengah, karena memiliki lebih banyak pilihan kelas instans yang mungkin memberikan opsi yang lebih hemat biaya dengan mengorbankan set fitur Aurora yang lebih kuat.

Model partisi SaaS multi-penyewa untuk PostgreSQL

Metode terbaik untuk mencapai multi-tenancy tergantung pada persyaratan untuk aplikasi SaaS Anda. Bagian berikut menunjukkan model partisi untuk berhasil menerapkan multi-tenancy di PostgreSQL.

> **Note**
>
> Model yang dibahas di bagian ini berlaku untuk Amazon RDS for PostgreSQL dan Aurora PostgreSQL yang kompatibel. Referensi ke PostgreSQL di bagian ini berlaku untuk kedua layanan.

Ada tiga model tingkat tinggi yang dapat Anda gunakan di PostgreSQL untuk partisi SaaS: silo, bridge, dan pool. Gambar berikut merangkum trade-off antara model silo dan pool. Model jembatan adalah hibrida dari model silo dan kolam renang.

Model partisi	Keuntungan	Kekurangan
Silo	• Keselarasan kepatuhan • Tidak ada dampak lintas penyewa • Penyetelan tingkat penyewa • Ketersediaan tingkat penyewa	• Kelincahan yang dikompromikan • Tidak ada manajemen terpusat pusat • Kompleksitas penyebaran • Biaya
Kolam	• Kelincahan • Optimalisasi biaya • Manajemen terpusat • Penyebaran sederhana	• Dampak lintas penyewa • Tantangan Kepatuhan • Semua atau tidak ada ketersediaan
Jembatan	• Beberapa kepatuhan • Kelincahan • Optimalisasi biaya	• Beberapa tantangan • Semua atau tidak ada ketersediaan (kebanyakan) • Dampak lintas penyewa

Model partisi	Keuntungan	Kekurangan
	• Manajemen terpusat	• Kompleksitas penyebaran

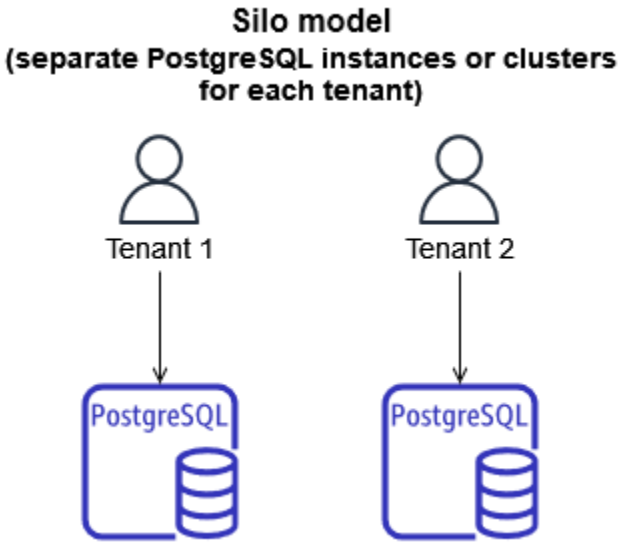
Bagian berikut membahas setiap model secara lebih mendetail.

Model partisi:

- Model silo PostgreSQL
- Model kolam renang PostgreSQL
- Model jembatan PostgreSQL
- Matriks keputusan

Model silo PostgreSQL

Model silo diimplementasikan dengan menyediakan instance PostgreSQL untuk setiap penyewa dalam aplikasi. Model silo unggul dalam kinerja penyewa dan isolasi keamanan, dan sepenuhnya menghilangkan fenomena tetangga yang bising. Fenomena tetangga yang bising terjadi ketika penggunaan sistem satu penyewa mempengaruhi kinerja penyewa lain. Model silo memungkinkan Anda menyesuaikan kinerja secara khusus untuk setiap penyewa dan berpotensi membatasi pemadaman pada silo penyewa tertentu. Namun, apa yang umumnya mendorong adopsi model silo adalah keamanan yang ketat dan kendala peraturan. Kendala ini dapat dimotivasi oleh pelanggan SaaS. Misalnya, pelanggan SaaS mungkin menuntut agar data mereka diisolasi karena kendala internal, dan penyedia SaaS mungkin menawarkan layanan semacam itu dengan biaya tambahan.

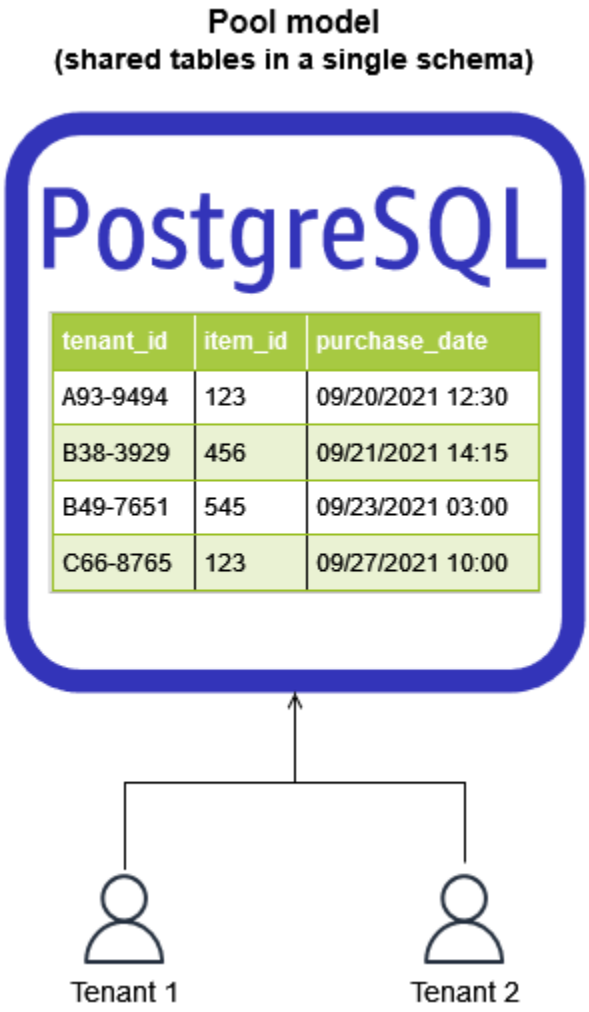


Meskipun model silo mungkin diperlukan dalam kasus-kasus tertentu, ia memiliki banyak kelemahan. Seringkali sulit untuk menggunakan model silo dengan cara yang hemat biaya, karena mengelola konsumsi sumber daya di beberapa instance PostgreSQL bisa jadi rumit. Selain itu, sifat terdistribusi beban kerja database dalam model ini membuatnya lebih sulit untuk mempertahankan pandangan terpusat dari aktivitas penyewa. Mengelola begitu banyak beban kerja yang dioperasikan secara independen meningkatkan biaya operasional dan administrasi. Model silo juga membuat orientasi penyewa lebih rumit dan memakan waktu, karena Anda harus menyediakan sumber daya khusus penyewa. Selain itu, seluruh sistem SaaS bisa lebih sulit untuk diskalakan, karena jumlah instance PostgreSQL khusus penyewa yang terus meningkat akan menuntut lebih banyak waktu operasional untuk dikelola. Satu pertimbangan terakhir adalah bahwa aplikasi atau lapisan akses data harus mempertahankan pemetaan penyewa ke instance PostgreSQL terkait mereka, yang menambah kompleksitas penerapan model ini.

Model kolam renang PostgreSQL

Model pangkalan diimplementasikan dengan menyediakan instans PostgreSQL tunggal (Amazon RDS atau Aurora) dan menggunakan keamanan tingkat baris (RLS) untuk mempertahankan isolasi data penyewa. Kebijakan RLS membatasi baris mana dalam tabel dikembalikan oleh`SELECT` kueri atau baris mana yang dipengaruhi oleh`INSERT`,`UPDATE`, dan`DELETE` perintah. Model pool memusatkan semua data penyewa dalam skema PostgreSQL tunggal, sehingga secara signifikan lebih hemat biaya dan membutuhkan lebih sedikit overhead operasional untuk mempertahankan. Pemantauan solusi ini juga secara signifikan lebih sederhana karena sentralisasi. Namun, memantau dampak spesifik penyewa dalam model kolam biasanya memerlukan beberapa instrumentasi

tambahan dalam aplikasi. Ini karena PostgreSQL secara default tidak mengetahui penyewa mana yang mengonsumsi sumber daya. Orientasi penyewa disederhanakan karena tidak ada infrastruktur baru yang diperlukan. Kelincahan ini membuatnya lebih mudah untuk mencapai alur kerja orientasi penyewa yang cepat dan otomatis.



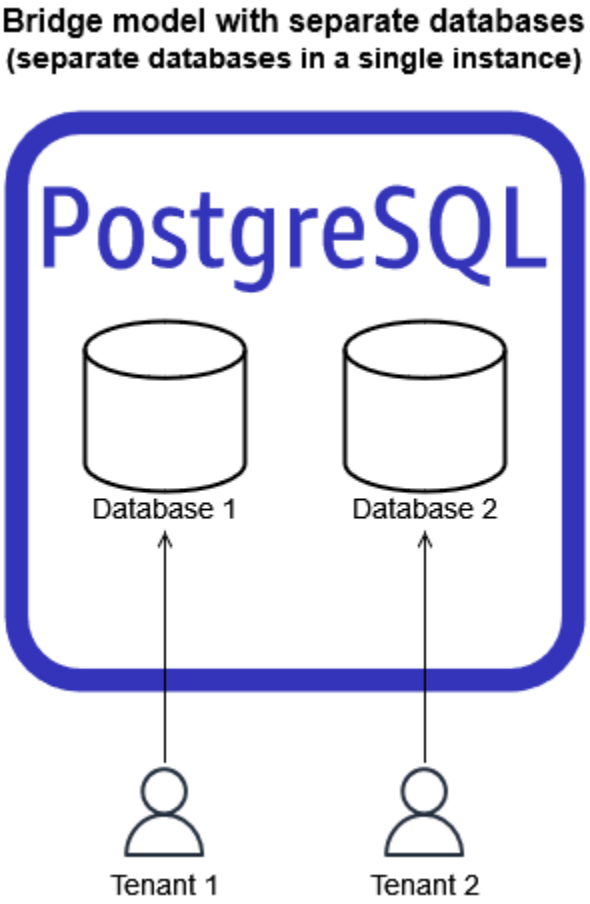
Meskipun model kolam renang umumnya lebih hemat biaya dan lebih sederhana untuk dikelola, ia memiliki beberapa kelemahan. Fenomena tetangga yang bising tidak dapat sepenuhnya dihilangkan dalam model kolam renang. Namun, ini dapat dikurangi dengan memastikan bahwa sumber daya yang sesuai tersedia pada instans PostgreSQL dan dengan menggunakan strategi untuk mengurangi beban di PostgreSQL, seperti membongkar kueri untuk membaca replika atau ke Amazon ElastiCache. Pemantauan yang efektif juga berperan dalam menanggapi masalah isolasi kinerja penyewa, karena instrumentasi aplikasi dapat mencatat dan memantau aktivitas khusus

penyewa. Terakhir, beberapa pelanggan SaaS mungkin tidak menemukan pemisahan logis yang disediakan oleh RLS cukup dan mungkin meminta tindakan isolasi tambahan.

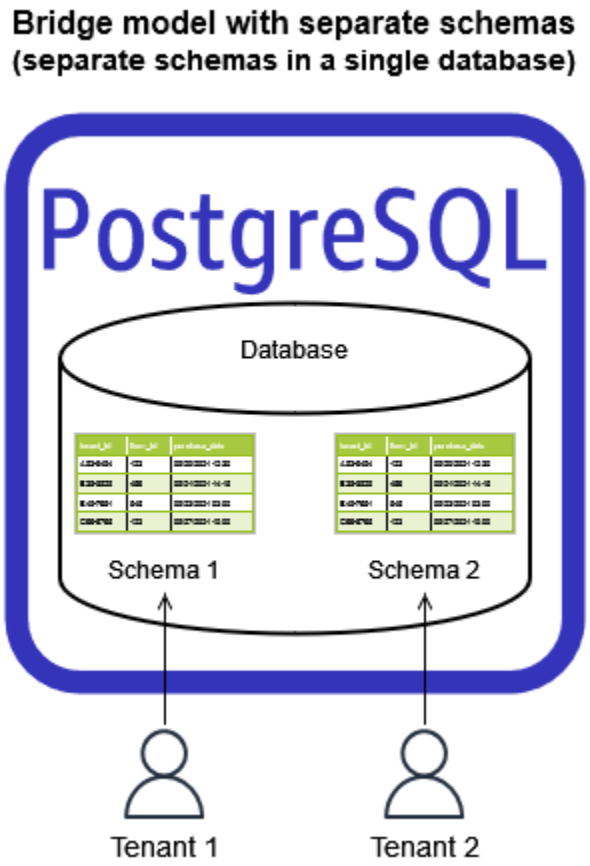
Model jembatan PostgreSQL

Model jembatan PostgreSQL adalah kombinasi dari pendekatan yang dikumpulkan dan dibungkam. Seperti model yang dikumpulkan, Anda menyediakan satu instance PostgreSQL untuk setiap penyewa. Untuk mempertahankan isolasi data penyewa, Anda menggunakan konstruksi logis PostgreSQL. Dalam diagram berikut, database PostgreSQL digunakan untuk memisahkan data secara logis.

> **Note**
>
> Database PostgreSQL tidak merujuk ke Amazon RDS terpisah untuk instans DB yang kompatibel dengan PostgreSQL atau Aurora PostgreSQL. Sebaliknya, ini mengacu pada konstruksi logis dari sistem manajemen database PostgreSQL untuk memisahkan data.



Anda juga dapat menerapkan model bridge dengan menggunakan database PostgreSQL tunggal, dengan skema khusus penyewa di setiap database, seperti yang diilustrasikan dalam diagram berikut.



Model jembatan menderita tetangga berisik yang sama dan masalah isolasi kinerja penyewa sebagai model kolam renang. Ini juga menimbulkan beberapa overhead operasional dan penyediaan tambahan dengan mengharuskan database atau skema terpisah untuk disediakan berdasarkan per penyewa. Diperlukan pemantauan yang efektif untuk merespons masalah kinerja penyewa dengan cepat. Ini juga membutuhkan instrumentasi aplikasi untuk memantau penggunaan khusus penyewa. Secara keseluruhan, model bridge dapat dilihat sebagai alternatif untuk RLS yang sedikit menambah upaya orientasi penyewa dengan membutuhkan database atau skema PostgreSQL baru. Seperti model silo, aplikasi atau lapisan akses data harus mempertahankan pemetaan penyewa ke database atau skema PostgreSQL terkait mereka.

Matriks keputusan

Untuk memutuskan model partisi SaaS multi-penyewa mana yang harus Anda gunakan dengan PostgreSQL, lihat matriks keputusan berikut. Matriks menganalisis empat opsi partisi ini:

- Silo - Instance atau klaster PostgreSQL terpisah untuk setiap penyewa.

- Jembatani dengan database terpisah - Database terpisah untuk setiap penyewa dalam satu instance atau klaster PostgreSQL.
- Jembatan dengan skema terpisah - Skema terpisah untuk setiap penyewa dalam satu database PostgreSQL, dalam satu instance atau klaster PostgreSQL.
- Pool - Tabel bersama untuk penyewa dalam satu contoh dan skema.

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Kasus penggunaan	Isolasi data dengan kontrol penuh atas penggunaan sumber daya adalah persyaratan utama, atau Anda memiliki penyewa yang sangat besar dan sangat sensitif terhadap kinerja.	Isolasi data adalah persyaratan utama, dan terbatas atau tidak ada referensi silang data penyewa diperlukan.	Sedang jumlah penyewa dengan jumlah moderat data. Ini adalah model yang disukai jika Anda harus referensi silang data penyewa.	Sejumlah besar penyewa dengan lebih sedikit data per penyewa.
Kelincahan orientasi penyewa baru	Sangat lambat. (Instance atau cluster baru diperlukan untuk setiap penyewa.)	Cukup lambat. (Membutuhkan pembuatan database baru untuk setiap penyewa untuk menyimpan objek skema.)	Cukup lambat. (Membutuhkan pembuatan skema baru untuk setiap penyewa untuk menyimpan objek.)	Opsi tercepat. (Diperlukan pengaturan minimal.)

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Koneksi database upaya konfigurasi kolam renang dan efisiensi	Diperlukan upaya yang signifikan. (Satu kolam koneksi per penyewa.) Kurang efisien. (Tidak ada koneksi database berbagi antara penyewa.)	Diperlukan upaya yang signifikan. (Satu konfigurasi kumpulan koneksi per penyewa kecuali Anda menggunakan Amazon RDS Proxy.) Kurang efisien. (Tidak ada koneksi database berbagi antara penyewa dan jumlah total koneksi. Penggunaan di semua penyewa dibatasan berdasarkan kelas instans DB.)	Kurang usaha yang dibutuhkan. (Satu konfigurasi kolam koneksi untuk semua penyewa.) Cukup efisien. (Koneksi digunakan kembali melaluiSET SCHEMA perintahSET ROLE atau dalam mode sesi kolam renang saja. SETperintah juga menyebabkan sesi menyematkan saat menggunakan Amazon RDS Proxy, tetapi kumpulan koneksi klien dapat dihilangkan dan koneksi langsung dapat dibuat untuk	Setidaknya usaha yang dibutuhkan. Paling efisien. (Satu kolam koneksi untuk semua penyewa dan penggunaan kembali koneksi yang efisien di semua penyewa. Batasan koneksi didasarkan pada kelas instans DB.)

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
			setiap permintaan efisiensi.)	
Pemeliharaan database (manajemen vakum) dan penggunaan sumber daya	Manajemen yang lebih sederhana.	kompleksitas sedang. (Mungkin menyebabkan konsumsi sumber daya yang tinggi, karena pekerja vakum harus dimulai untuk setiap database setelahnya `avacuum_naptime` , yang mengarah ke penggunaan CPU peluncur autovacuum tinggi. Mungkin juga ada overhead tambahan yang terkait dengan penyedotan tabel katalog sistem PostgreSQL untuk setiap database.)	Tabel katalog sistem PostgreSQL besar. (`pg_catalog` Ukuran total dalam puluhan GB, tergantung pada jumlah penyewa dan relasi. Mungkin memerlukan modifikasi parameter yang berhubungan dengan debu untuk mengontrol meja mengasapi.)	Tabel mungkin besar, tergantung pada jumlah penyewa dan data per penyewa. (Kemungkinan memerlukan modifikasi parameter terkait debu untuk mengontrol mengasapi meja.)

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Upaya manajemen ekstensi	Upaya signifikan (untuk setiap database dalam kasus terpisah).	Upaya yang signifikan (di setiap tingkat database).	Upaya minimal (satu kali dalam database umum).	Upaya minimal (satu kali dalam database umum).
Ubah upaya penyebaran	Upaya signifikan. (Connect ke setiap instance terpisah dan luncurkan perubahan.)	Upaya signifikan. (Connect ke setiap database dan skema, dan meluncurkan perubahan.)	Upaya moderat. (Connect ke database umum dan meluncurkan perubahan untuk setiap skema.)	Upaya minimal. (Connect ke database umum dan meluncurkan perubahan.)
Ubah penyebaran — cakupan dampak	minimal minimal. (Penyewa tunggal terpengaruh.)	minimal minimal. (Penyewa tunggal terpengaruh.)	minimal minimal. (Penyewa tunggal terpengaruh.)	Sangat besar. (Semua penyewa terpengaruh.)

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Manajemen dan upaya kinerja kueri	Kinerja query dikelola.	Kinerja query dikelola.	Kinerja query dikelola.	Upaya signifikan kemungkinan diperlukan untuk mempertahankan kinerja kueri. (Seiring waktu, kueri mungkin berjalan lebih lambat karena peningkatan ukuran tabel. Anda dapat menggunakan partisi tabel dan sharding database untuk mempertahankan kinerja.)
Dampak sumber daya lintas penyewa	Tidak ada Dampak. (Tidak ada berbagi sumber daya di antara penyewa.)	Dampak moderat. (Penyewa berbagi sumber daya umum seperti misalnya CPU dan memori.)	Dampak moderat. (Penyewa berbagi sumber daya umum seperti misalnya CPU dan memori.)	Dampak berat. (Penyewa saling mempengaruhi dalam hal sumber daya, mengunci konflik, dan sebagainya.)

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Penyewa tingkat tuning (misalnya, penciptaan indeks tambahan per penyewa atau parameter DB tweaking untuk penyewa tertentu)	Kemungkinan.	Agak mungkin. (Perubahan tingkat skema dapat dilakukan untuk setiap penyewa, tetapi parameter database bersifat global di semua penyewa.)	Agak mungkin. (Perubahan tingkat skema dapat dilakukan untuk setiap penyewa, tetapi parameter database bersifat global di semua penyewa.)	Tidak memungkinkan. (Tabel dibagi oleh semua penyewa.)

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Upaya menyeimbangkan kembali penyewa yang peka terhadap kinerja	minimal minimal. (Tidak perlu menyeimbangkan kembali. Skala server dan sumber daya I/O untuk menangani skenario ini.)	Sedang. (Gunakan replikasi logis ataupg_dump untuk mengekspor database, tetapi downtime mungkin panjang tergantung pada ukuran data. Anda dapat menggunakan fitur database yang dapat diangkut di Amazon RDS for PostgreSQL untuk menyalin database antar instans dengan lebih cepat.)	Sedang tetapi kemungkinan melibatkan waktu henti yang lama. (Gunakan replikasi logis ataupg_dump untuk mengekspor skema, tetapi downtime mungkin panjang tergantung pada ukuran data.)	Signifikan, karena semua penyewa berbagi tabel yang sama. (Sharding database membutuhkan menyalin semuanya ke instance lain dan langkah tambahan untuk membersihkan data penyewa.) Kemungkinan besar membutuhkan perubahan logika aplikasi.

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Downtime database untuk peningkatan versi mayor	Downtime standar. (Tergantung pada ukuran katalog sistem PostgreSQL.)	Kemungkinan downtime yang lebih lama. (Tergantung pada ukuran katalog sistem, waktu akan bervariasi. Tabel katalog sistem PostgreSQL juga diduplikasi di seluruh database)	Kemungkinan downtime yang lebih lama. (Tergantung pada ukuran katalog sistem PostgreSQL, waktu akan bervariasi.)	Downtime standar. (Tergantung pada ukuran katalog sistem PostgreSQL.)
Overhead administrasi (misalnya, untuk analisis log database atau pemantauan pekerjaan cadangan)	Upaya signifikan	Upaya minimal.	Upaya minimal.	Upaya minimal.

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Ketersediaan tingkat penyewa	Tertinggi. (Setiap penyewa gagal dan pulih secara independen.)	Lingkup dampak yang lebih tinggi. (Semua penyewa gagal dan pulih bersama jika terjadi masalah perangkat keras atau sumber daya.)	Lingkup dampak yang lebih tinggi. (Semua penyewa gagal dan pulih bersama jika terjadi masalah perangkat keras atau sumber daya.)	Lingkup dampak yang lebih tinggi. (Semua penyewa gagal dan pulih bersama jika terjadi masalah perangkat keras atau sumber daya.)
Upaya pencadangan dan pemulihan tingkat penyewa	Upaya paling tidak tertunda. (Setiap penyewa dapat didukung dan dipulihkan secara independen.)	Upaya moderat. (Gunakan ekspor logis dan impor untuk setiap penyewa. Beberapa pengkodean dan otomatisasi diperlukan.)	Upaya moderat. (Gunakan ekspor logis dan impor untuk setiap penyewa. Beberapa pengkodean dan otomatisasi diperlukan.)	Upaya signifikan. (Semua penyewa berbagi tabel yang sama.)
Upaya point-in-time pemulihan tingkat penyewa	Upaya minimal. (Gunakan pemulihan waktu point-in dengan menggunakan snapshot, atau gunakan pelacakan mundur di Amazon Aurora.)	Upaya moderat. (Gunakan snapshot restore, diikuti oleh ekspor/impor. Namun, ini akan menjadi operasi yang lambat.)	Upaya moderat. (Gunakan snapshot restore, diikuti oleh ekspor/impor. Namun, ini akan menjadi operasi yang lambat.)	Upaya dan kompleksitas yang signifikan.

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Nama skema seragam	Nama skema yang sama untuk setiap penyewa.	Nama skema yang sama untuk setiap penyewa.	Skema yang berbeda untuk setiap penyewa.	Skema umum.
Kustomisasi per penyewa (misalnya, kolom tabel tambahan untuk penyewa tertentu)	Kemungkinan.	Kemungkinan.	Kemungkinan.	Rumit (karena semua penyewa berbagi tabel yang sama).
Efisiensi manajemen katalog pada layer object-relasional mapping (ORM) (misalnya, Ruby)	Efisien (karena koneksi klien khusus untuk penyewa).	Efisien (karena koneksi klien khusus untuk database).	Cukup efisien. (Bergantung pada ORM yang digunakan, model keamanan pengguna/peran, dan `search_path` konfigurasi, klien terkadang menyimpan metadata untuk semua penyewa, yang mengarah ke penggunaan memori koneksi DB yang tinggi.)	Efisien (karena semua penyewa berbagi tabel yang sama).

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Upaya pelaporan penyewa konsolidasi	Upaya signifikan. (Anda harus menggunakan pembungkus data asing [FDWs] untuk mengkonsolidasikan data di semua penyewa atau mengekstrak, mengubah, dan memuat [ETL] ke database pelaporan lain.)	Upaya signifikan. (Anda harus menggunakan FDW untuk mengkonsolidasikan data di semua penyewa atau ETL ke database pelaporan lain.)	Upaya moderat. (Anda dapat mengumpulkan data dalam semua skema dengan menggunakan serikat pekerja.)	Upaya minimal. (Semua data penyewa ada dalam tabel yang sama, jadi pelaporan itu sederhana.)
Instance hanya-baca khusus penyewa untuk pelaporan (misalnya, berdasarkan langganan)	Upaya paling tidak tertunda. (Ciptakan sebuah replika baca baca.)	Upaya moderat. (Anda dapat menggunakan replikasi logis atauAWS Database Migration Service [AWS DMS] untuk mengkonfigurasi.)	Upaya moderat. (Anda dapat menggunakan replikasi logis atauAWS DMS untuk mengkonfigurasi.)	Rumit (karena semua penyewa berbagi tabel yang sama).

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Isolasi data	Terbaik.	Lebih baik. (Anda dapat mengelola izin tingkat database dengan menggunakan peran PostgreSQL.)	Lebih baik. (Anda dapat mengelola izin tingkat skema dengan menggunakan peran PostgreSQL.)	Lebih buruk. (Karena semua penyewa berbagi tabel yang sama, Anda harus menerapkan fitur seperti keamanan tingkat baris [RLS] untuk isolasi penyewa.)
Kunci enkripsi penyimpanan khusus penyewa	Kemungkinan. (Setiap klaster PostgreSQL dapat memiliki kunciAWS Key Management Service [AWS KMS] sendiri untuk enkripsi penyimpanan.)	Tidak memungkinkan. (Semua penyewa berbagi kunci KMS yang sama untuk enkripsi penyimpanan.)	Tidak memungkinkan. (Semua penyewa berbagi kunci KMS yang sama untuk enkripsi penyimpanan.)	Tidak memungkinkan. (Semua penyewa berbagi kunci KMS yang sama untuk enkripsi penyimpanan.)
Menggunakan anAWS Identity and Access Management (IAM) untuk otentikasi database untuk setiap penyewa	Kemungkinan.	Kemungkinan.	Kemungkinan (dengan memiliki pengguna PostgreSQL terpisah untuk setiap skema).	Tidak mungkin (karena tabel dibagi oleh semua penyewa).

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Biaya infrastruktur	Tertinggi (karena tidak ada yang dibagikan).	Sedang.	Sedang.	Terendah.
Duplikasi data dan penggunaan penyimpanan	Agregat tertinggi di semua penyewa. (Tabel katalog sistem PostgreSQL dan data statis dan umum aplikasi diduplikasi di semua penyewa.)	Agregat tertinggi di semua penyewa. (Tabel katalog sistem PostgreSQL dan data statis dan umum aplikasi diduplikasi di semua penyewa.)	Sedang. (Data statis dan umum aplikasi dapat dalam skema umum dan diakses oleh penyewa lain.)	minimal minimal. (Tidak ada duplikasi data. Data statis dan umum aplikasi dapat dalam skema yang sama.)
Pemantauan yang berpusat pada penyewa (cari tahu penyewa mana yang menyebabkan masalah)	Upaya paling tidak tertunda. (Karena setiap penyewa dipantau secara terpisah, mudah untuk memeriksa aktivitas penyewa tertentu.)	Upaya moderat. (Karena semua penyewa berbagi sumber daya fisik yang sama, Anda harus menerapkan penyaringan tambahan untuk memeriksa aktivitas penyewa tertentu.)	Upaya moderat. (Karena semua penyewa berbagi sumber daya fisik yang sama, Anda harus menerapkan penyaringan tambahan untuk memeriksa aktivitas penyewa tertentu.)	Upaya signifikan. (Karena semua penyewa berbagi semua sumber daya, termasuk tabel, Anda harus menggunakan penangkapan variabel bind untuk memeriksa penyewa mana yang dimiliki oleh kueri SQL tertentu.)

	Silo	Jembatan dengan database terpisah	Jembatan dengan skema terpisah	Kolam
Manajemen terpusat dan pemantaua n kesehatan/ aktivitas	Upaya signifikan (untuk mengatur pemantauan pusat dan pusat komando pusat).	Upaya moderat (karena semua penyewa berbagi contoh yang sama).	Upaya moderat (karena semua penyewa berbagi contoh yang sama).	Upaya minimal (karena semua penyewa berbagi sumber daya yang sama, termasuk skema).
Kemungkinan obyek identifie r (OID) dan ID transaksi (XID) sampul	minimal minimal.	Tinggi. (Karena OID, XID adalah penghitun g klaster PostgreSQ L tunggal dan mungkin ada masalah penyedotan debu secara efektif di seluruh database fisik).	Sedang. (Karena OID, XID adalah penghitun g klaster PostgreSQL tunggal).	Tinggi. (Misalnya , satu tabel dapat mencapai batas TOAST OID sebesar 4 miliar, tergantung pada jumlah out-of-line kolom.)

Rekomendasi keamanan

Keamanan tingkat baris (RLS) diperlukan untuk menjaga isolasi data penyewa dalam model yang dikumpulkan dengan PostgreSQL. RLS memusatkan penegakan kebijakan isolasi di tingkat database dan menghilangkan beban mempertahankan isolasi ini dari pengembang perangkat lunak. Cara paling umum untuk mengimplementasikan RLS adalah mengaktifkan fitur ini di PostgreSQL DBMS. RLS melibatkan penyaringan akses ke baris data berdasarkan nilai dalam kolom tertentu. Anda dapat menggunakan dua metode untuk memfilter akses ke data:

- Kolom data tertentu dalam tabel dibandingkan dengan nilai pengguna PostgreSQL saat ini. Nilai di kolom yang setara dengan pengguna PostgreSQL yang masuk dapat diakses oleh pengguna tersebut.
- Kolom data tertentu dalam tabel dibandingkan dengan nilai variabel runtime yang ditetapkan oleh aplikasi. Nilai dalam kolom yang setara dengan variabel runtime dapat diakses selama sesi itu.

Opsi kedua lebih disukai, karena opsi pertama memerlukan pembuatan pengguna PostgreSQL baru untuk setiap penyewa. Sebagai gantinya, aplikasi SaaS yang menggunakan PostgreSQL harus bertanggung jawab untuk mengatur konteks khusus penyewa saat runtime saat meminta PostgreSQL. Ini akan memiliki efek menegakkan RLS. Anda juga dapat mengaktifkan RLS table-by-table secara. Sebagai praktik terbaik, Anda harus mengaktifkan RLS pada semua tabel yang berisi data penyewa.

Contoh berikut ini menampilkan dua tabel dan memungkinkan RLS. Contoh ini membandingkan kolom data dengan nilai variabel runtime`app.current_tenant`.

```
-- Create a table for our tenants with indexes on the primary key and the tenant’s name
CREATE TABLE tenant (
    tenant_id UUID DEFAULT uuid_generate_v4() PRIMARY KEY,
    name VARCHAR(255) UNIQUE,
    status VARCHAR(64) CHECK (status IN ('active', 'suspended', 'disabled')),
    tier VARCHAR(64) CHECK (tier IN ('gold', 'silver', 'bronze'))
);

-- Create a table for users of a tenant
CREATE TABLE tenant_user (
    user_id UUID DEFAULT uuid_generate_v4() PRIMARY KEY,
    tenant_id UUID NOT NULL REFERENCES tenant (tenant_id) ON DELETE RESTRICT,
    email VARCHAR(255) NOT NULL UNIQUE,
```

```
    given_name VARCHAR(255) NOT NULL CHECK (given_name <> ''),
    family_name VARCHAR(255) NOT NULL CHECK (family_name <> '')
);

-- Turn on RLS
ALTER TABLE tenant ENABLE ROW LEVEL SECURITY;

-- Restrict read and write actions so tenants can only see their rows
-- Cast the UUID value in tenant_id to match the type current_setting
-- This policy implies a WITH CHECK that matches the USING clause
CREATE POLICY tenant_isolation_policy ON tenant
USING (tenant_id = current_setting('app.current_tenant')::UUID);

-- And do the same for the tenant users
ALTER TABLE tenant_user ENABLE ROW LEVEL SECURITY;

CREATE POLICY tenant_user_isolation_policy ON tenant_user
USING (tenant_id = current_setting('app.current_tenant')::UUID);
```

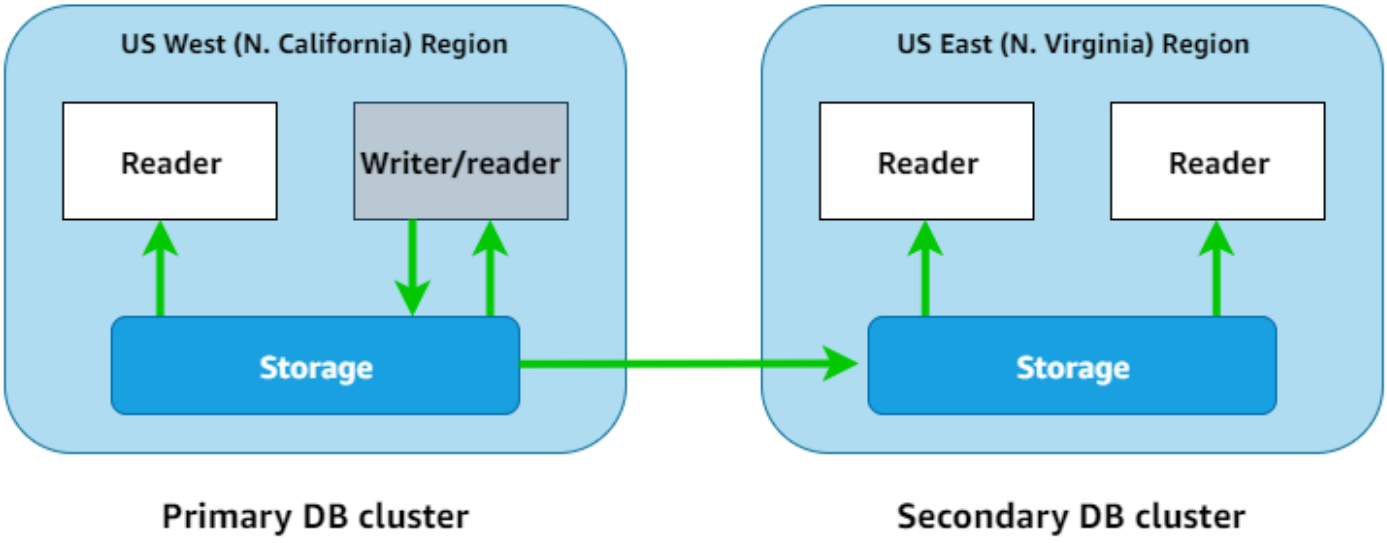
Untuk informasi lebih lanjut, lihat posting blog Isolasi data multi-penyewa dengan PostgreSQL tingkat TimAWS SaaS Factory juga memiliki beberapa contoh GitHub untuk membantu dalam menerapkan RLS.

Ketersediaan PostgreSQL untuk model kolam

Model kolam menurut sifatnya hanya memiliki satu instance PostgreSQL. Oleh karena itu, merancang aplikasi Anda untuk ketersediaan tinggi sangat penting. Kegagalan atau pemadaman database yang dikumpulkan mengakibatkan aplikasi Anda terdegradasi atau menjadi tidak dapat diakses oleh semua penyewa Anda.

Instans Amazon RDS for PostgreSQL DB dapat dibuat redundan di dua Availability Zone dengan mengaktifkan fitur ketersediaan tinggi. Untuk informasi selengkapnya, lihat Ketersediaan tinggi (Multi-AZ) untuk Amazon RDS dalam dokumentasi Amazon RDS. Untuk failover lintas wilayah, Anda dapat membuat replika baca di Wilayah yang berbeda. AWS (Replika baca ini harus dipromosikan sebagai bagian dari proses failover.) Selain itu, Anda dapat mereplikasi cadangan yang direplikasi di seluruh AWS Wilayah untuk pemulihan. Untuk informasi selengkapnya, lihat Mereplikasi pencadangan otomatis ke AWS Wilayah lain dalam dokumentasi Amazon RDS.

Aurora PostgreSQL kompatibel secara otomatis mencadangkan data dengan cara yang dapat mempertahankan kegagalan beberapa Availability Zone. (Lihat Ketersediaan tinggi untuk Amazon Aurora di dokumentasi Aurora.) Untuk membuat Aurora lebih tangguh dan pulih lebih cepat, Anda dapat membuat replika baca Aurora di Availability Zone lainnya. Anda dapat menggunakan database global Aurora untuk mereplikasi data menjadi lima AWS Wilayah tambahan untuk pemulihan Lintas wilayah dan failover otomatis. (Lihat Menggunakan database global Amazon Aurora dalam dokumentasi Aurora.) Selain itu, Anda dapat mengaktifkan penerusan tulis dengan basis data global Aurora untuk mencapai ketersediaan tinggi di beberapa. Wilayah AWS



Terlepas dari apakah Anda menggunakan Amazon RDS untuk PostgreSQL atau Aurora PostgreSQL yang kompatibel dengan Aurora, sebaiknya Anda menerapkan fitur ketersediaan tinggi untuk mengurangi dampak pemadaman apa pun untuk semua aplikasi SaaS multi-penyewa yang menggunakan model pool.

Praktik terbaik

Bagian ini mencantumkan beberapa takeaway tingkat tinggi dari panduan ini. Untuk diskusi terperinci tentang setiap titik, ikuti tautan ke bagian yang sesuai.

BandingkanAWS opsi untuk PostgreSQL terkelola

AWSmenawarkan dua cara utama untuk menjalankan PostgreSQL di lingkungan yang dikelola. (Dalam konteks ini, dikelola berarti bahwa infrastruktur PostgreSQL dan DBMS sebagian atau seluruhnya didukung olehAWS layanan.) Opsi PostgreSQL yang dikelolaAWS memiliki manfaat mengotomatisasi cadangan, failover, optimasi, dan beberapa administrasi PostgreSQL. Sebagai opsi Amazon Aurora Edisi Kompatibel PostgreSQL dan Amazon Relational Database Service Edisi Kompatibel PostgreSQL.AWS Anda dapat memilih pilihan terbaik dari kedua model ini dengan menganalisis kasus penggunaan PostgreSQL Anda. Untuk informasi selengkapnya, lihat bagian Memilih antara Amazon RDS dan Aurora dalam panduan ini.

Pilih model partisi SaaS multi-penyewa

Anda dapat memilih dari tiga model partisi SaaS yang berlaku untuk PostgreSQL: silo, bridge, dan pool. Setiap model memiliki kelebihan dan kekurangan, dan Anda harus memilih model yang paling optimal tergantung pada kasus penggunaan Anda. Amazon RDS for PostgreSQL dan Aurora PostgreSQL yang kompatibel mendukung ketiga model tersebut. Memilih model sangat penting untuk menjaga isolasi data penyewa dalam aplikasi SaaS Anda. Untuk diskusi mendetail tentang model-model ini, lihat bagian Model partisi SaaS multi-tenant untuk PostgreSQL dalam panduan ini.

Gunakan keamanan tingkat baris untuk model partisi SaaS kolam renang

Keamanan tingkat baris (RLS) diperlukan untuk menjaga isolasi data penyewa dalam model pool dengan PostgreSQL. Hal ini karena tidak ada pemisahan logis antara infrastruktur, database PostgreSQL, atau skema pada basis per-penyewa dalam model pool. RLS memusatkan penegakan kebijakan isolasi di tingkat database dan menghilangkan beban mempertahankan isolasi ini dari pengembang perangkat lunak. Anda dapat menggunakan RLS untuk membatasi operasi database untuk penyewa tertentu. Untuk informasi selengkapnya dan contoh, lihat bagian Rekomendasi keamanan tingkat baris di panduan ini.

Pertanyaan yang Sering Diajukan

Bagian ini memberikan jawaban atas pertanyaan umum tentang penerapan PostgreSQL terkelola dalam aplikasi SaaS multi-penyewa.

Opsi PostgreSQL terkelola mana yangAWS ditawarkan?

AWSmenawarkan Amazon Aurora PostgreSQL yang kompatibel dengan Amazon dan Amazon Relational Database Service (Amazon RDS) untuk PostgreSQL. AWSjuga memiliki katalog luas penawaran database terkelola.

Layanan mana yang optimal untuk aplikasi SaaS?

Anda dapat menggunakan aplikasi Aurora PostgreSQL dan Amazon RDS for PostgreSQL untuk aplikasi SaaS dan semua model partisi SaaS yang dibahas dalam panduan ini. Kedua layanan ini memiliki perbedaan dalam skalabilitas, pemulihan kerusakan, failover, opsi penyimpanan, ketersediaan tinggi, pemulihan bencana, cadangan, dan kelas instans yang tersedia untuk setiap opsi. Pilihan optimal akan tergantung pada kasus penggunaan spesifik Anda. Gunakan matriks keputusan dalam panduan ini untuk memilih opsi terbaik untuk kasus penggunaan Anda.

Persyaratan unik mana yang harus saya pertimbangkan jika saya memutuskan untuk menggunakan database PostgreSQL dengan aplikasi SaaS multi-tenant?

Seperti halnya penyimpanan data yang digunakan dengan aplikasi SaaS, pertimbangan yang paling penting adalah metode untuk menjaga isolasi data penyewa. Seperti yang dibahas dalam panduan ini, ada beberapa cara untuk mencapai isolasi data penyewa dengan penawaran PostgreSQL yangAWS dikelola. Selain itu, Anda harus mempertimbangkan isolasi kinerja berdasarkan per-tenant untuk implementasi PostgreSQL apa pun.

Model mana yang dapat saya gunakan untuk mempertahankan isolasi data penyewa dengan PostgreSQL?

Anda dapat menggunakan model silo, bridge, dan pool sebagai strategi partisi SaaS untuk mempertahankan isolasi data penyewa. Untuk diskusi tentang model-model ini dan bagaimana

mereka dapat diterapkan ke PostgreSQL, lihat bagian Model partisi SaaS multi-tenant untuk PostgreSQL dalam panduan ini.

Bagaimana cara menjaga isolasi data penyewa dengan satu database PostgreSQL yang dibagikan di beberapa penyewa?

PostgreSQL mendukung fitur keamanan tingkat baris (RLS) yang dapat Anda gunakan untuk menegakkan isolasi data penyewa dalam satu database atau instance PostgreSQL. Selain itu, Anda dapat menyediakan database PostgreSQL terpisah per penyewa dalam satu contoh, atau membuat skema berdasarkan per penyewa untuk mencapai tujuan ini. Untuk kelebihan dan kekurangan pendekatan ini, lihat bagian Rekomendasi keamanan tingkat baris dalam panduan ini.

Langkah selanjutnya

AWSmenawarkan dua opsi untuk mengoperasikan PostgreSQL terkelola: Kompatibel dengan Aurora PostgreSQL dan Amazon RDS for PostgreSQL. Kami menyarankan Anda mengevaluasi kedua layanan dan memilih opsi yang paling mendukung kasus penggunaan spesifik Anda untuk aplikasi SaaS multi-penyewa Anda. Sesuai dengan model partisi SaaS dapat memastikan bahwa aplikasi SaaS yang menggunakan PostgreSQL mematuhi secara ketat praktik terbaik untuk mempertahankan sewa. Model partisi silo, jembatan, dan kolam SaaS mendukung banyak kasus penggunaan SaaS. Model ini memberikan berbagai keuntungan di antara faktor-faktor seperti isolasi kinerja, overhead operasional, dan keamanan penyewa.

Langkah Selanjutnya yang disamping amping amping

- Evaluasi Aurora PostgreSQL yang kompatibel dan Amazon RDS for PostgreSQL, dan pilih opsi terbaik untuk aplikasi SaaS Anda.
- Pilih model partisi SaaS yang memenuhi persyaratan untuk aplikasi Anda: silo, jembatan, atau kolam renang.
- Menerapkan PostgreSQL sesuai dengan model partisi SaaS yang Anda pilih.

Sumber daya

Referensi

- Strategi Penyimpanan SaaS: Membangun Model Penyimpanan Multitenant di (whitepaper) AWS AWS
- Pemulihan bencana Lintas Wilayah menggunakan Amazon Aurora Global Database untuk Amazon Aurora PostgreSQL (posting blog) AWS
- Isolasi data multitenant dengan PostgreSQL Row Level Security (posting blog) AWS
- Bekerja dengan Amazon Aurora PostgreSQL (dokumentasi Aurora)
- PostgreSQL di Amazon RDS (dokumentasi Amazon RDS)

Mitra

- Amazon Aurora untuk Mitra PostgreSQL
- Amazon RDS untuk Mitra PostgreSQL

Riwayat dokumen

Tabel berikut menjelaskan perubahan signifikan pada panduan ini. Jika Anda ingin diberi tahu tentang pembaruan masa depan, Anda dapat berlangganan umpan RSS.

Perubahan	Deskripsi	Tanggal
Perbarui	Pembaruan untuk mencerminkan ketersediaan penerusan tulis di Aurora.	April 29, 2024
Perbarui	Memperbarui tabel perbandingan Amazon RDS dan Aurora.	21 Oktober 2022
—	Publikasi awal	30 September 2021

AWS Glosarium Panduan Preskriptif

Berikut ini adalah istilah yang umum digunakan dalam strategi, panduan, dan pola yang disediakan oleh Panduan AWS Preskriptif. Untuk menyarankan entri, silakan gunakan tautan Berikan umpan balik di akhir glosarium.

Nomor

7 Rs

Tujuh strategi migrasi umum untuk memindahkan aplikasi ke cloud. Strategi ini dibangun di atas 5 Rs yang diidentifikasi Gartner pada tahun 2011 dan terdiri dari yang berikut:

- Refactor/Re-Architect — Memindahkan aplikasi dan memodifikasi arsitekturnya dengan memanfaatkan sepenuhnya fitur cloud-native untuk meningkatkan kelincahan, kinerja, dan skalabilitas. Ini biasanya melibatkan porting sistem operasi dan database. Contoh: Migrasikan database Oracle lokal Anda ke Amazon Aurora PostgreSQL Compatible Edition.
- Replatform (angkat dan bentuk ulang) — Pindahkan aplikasi ke cloud, dan perkenalkan beberapa tingkat pengoptimalan untuk memanfaatkan kemampuan cloud. Contoh: Migrasikan database Oracle lokal Anda ke Amazon Relational Database Service (Amazon RDS) untuk Oracle di. AWS Cloud
- Pembelian kembali (drop and shop) - Beralih ke produk yang berbeda, biasanya dengan beralih dari lisensi tradisional ke model SaaS. Contoh: Migrasikan sistem manajemen hubungan pelanggan (CRM) Anda ke Salesforce.com.
- Rehost (lift dan shift) — Pindahkan aplikasi ke cloud tanpa membuat perubahan apa pun untuk memanfaatkan kemampuan cloud. Contoh: Migrasikan database Oracle lokal Anda ke Oracle pada instans EC2 di. AWS Cloud
- Relokasi (hypervisor-level lift and shift) — Pindahkan infrastruktur ke cloud tanpa membeli perangkat keras baru, menulis ulang aplikasi, atau memodifikasi operasi yang ada. Anda memigrasikan server dari platform lokal ke layanan cloud untuk platform yang sama. Contoh: Migrasikan Microsoft Hyper-V aplikasi ke AWS.
- Pertahankan (kunjungi kembali) - Simpan aplikasi di lingkungan sumber Anda. Ini mungkin termasuk aplikasi yang memerlukan refactoring besar, dan Anda ingin menunda pekerjaan itu sampai nanti, dan aplikasi lama yang ingin Anda pertahankan, karena tidak ada pembenaran bisnis untuk memigrasikannya.

- Pensiun — Menonaktifkan atau menghapus aplikasi yang tidak lagi diperlukan di lingkungan sumber Anda.

A

ABAC

Lihat kontrol akses berbasis atribut.

layanan abstrak

Lihat layanan terkelola.

ASAM

Lihat atomisitas, konsistensi, isolasi, daya tahan.

migrasi aktif-aktif

Metode migrasi database di mana database sumber dan target tetap sinkron (dengan menggunakan alat replikasi dua arah atau operasi penulisan ganda), dan kedua database menangani transaksi dari menghubungkan aplikasi selama migrasi. Metode ini mendukung migrasi dalam batch kecil yang terkontrol alih-alih memerlukan pemotongan satu kali. Ini lebih fleksibel tetapi membutuhkan lebih banyak pekerjaan daripada migrasi aktif-pasif.

migrasi aktif-pasif

Metode migrasi database di mana database sumber dan target disimpan dalam sinkron, tetapi hanya database sumber yang menangani transaksi dari menghubungkan aplikasi sementara data direplikasi ke database target. Basis data target tidak menerima transaksi apa pun selama migrasi.

fungsi agregat

Fungsi SQL yang beroperasi pada sekelompok baris dan menghitung nilai pengembalian tunggal untuk grup. Contoh fungsi agregat meliputi `SUM` dan`MAX`.

AI

Lihat kecerdasan buatan.

AIOps

Lihat operasi kecerdasan buatan.

anonimisasi

Proses menghapus informasi pribadi secara permanen dalam kumpulan data. Anonimisasi dapat membantu melindungi privasi pribadi. Data anonim tidak lagi dianggap sebagai data pribadi.

anti-pola

Solusi yang sering digunakan untuk masalah berulang di mana solusinya kontra-produktif, tidak efektif, atau kurang efektif daripada alternatif.

kontrol aplikasi

Pendekatan keamanan yang memungkinkan penggunaan hanya aplikasi yang disetujui untuk membantu melindungi sistem dari malware.

portofolio aplikasi

Kumpulan informasi rinci tentang setiap aplikasi yang digunakan oleh organisasi, termasuk biaya untuk membangun dan memelihara aplikasi, dan nilai bisnisnya. Informasi ini adalah kunci untuk penemuan portofolio dan proses analisis dan membantu mengidentifikasi dan memprioritaskan aplikasi yang akan dimigrasi, dimodernisasi, dan dioptimalkan.

kecerdasan buatan (AI)

Bidang ilmu komputer yang didedikasikan untuk menggunakan teknologi komputasi untuk melakukan fungsi kognitif yang biasanya terkait dengan manusia, seperti belajar, memecahkan masalah, dan mengenali pola. Untuk informasi lebih lanjut, lihat Apa itu Kecerdasan Buatan?

operasi kecerdasan buatan (AIOps)

Proses menggunakan teknik pembelajaran mesin untuk memecahkan masalah operasional, mengurangi insiden operasional dan intervensi manusia, dan meningkatkan kualitas layanan. Untuk informasi selengkapnya tentang cara AIOps digunakan dalam strategi AWS migrasi, lihat panduan integrasi operasi.

enkripsi asimetris

Algoritma enkripsi yang menggunakan sepasang kunci, kunci publik untuk enkripsi dan kunci pribadi untuk dekripsi. Anda dapat berbagi kunci publik karena tidak digunakan untuk dekripsi, tetapi akses ke kunci pribadi harus sangat dibatasi.

atomisitas, konsistensi, isolasi, daya tahan (ACID)

Satu set properti perangkat lunak yang menjamin validitas data dan keandalan operasional database, bahkan dalam kasus kesalahan, kegagalan daya, atau masalah lainnya.

kontrol akses berbasis atribut (ABAC)

Praktik membuat izin berbutir halus berdasarkan atribut pengguna, seperti departemen, peran pekerjaan, dan nama tim. Untuk informasi selengkapnya, lihat ABAC untuk AWS dokumentasi AWS Identity and Access Management (IAM).

sumber data otoritatif

Lokasi di mana Anda menyimpan versi utama data, yang dianggap sebagai sumber informasi yang paling dapat diandalkan. Anda dapat menyalin data dari sumber data otoritatif ke lokasi lain untuk tujuan memproses atau memodifikasi data, seperti menganonimkan, menyunting, atau membuat nama samaran.

Zona Ketersediaan

Lokasi berbeda di dalam Wilayah AWS yang terisolasi dari kegagalan di Availability Zone lainnya dan menyediakan konektivitas jaringan latensi rendah yang murah ke Availability Zone lainnya di Wilayah yang sama.

AWS Kerangka Adopsi Cloud (AWS CAF)

Kerangka pedoman dan praktik terbaik AWS untuk membantu organisasi mengembangkan rencana yang efisien dan efektif untuk bergerak dengan sukses ke cloud. AWS CAF mengatur panduan ke dalam enam area fokus yang disebut perspektif: bisnis, orang, tata kelola, platform, keamanan, dan operasi. Perspektif bisnis, orang, dan tata kelola fokus pada keterampilan dan proses bisnis; perspektif platform, keamanan, dan operasi fokus pada keterampilan dan proses teknis. Misalnya, perspektif masyarakat menargetkan pemangku kepentingan yang menangani sumber daya manusia (SDM), fungsi kepegawaian, dan manajemen orang. Untuk perspektif ini, AWS CAF memberikan panduan untuk pengembangan, pelatihan, dan komunikasi orang untuk membantu mempersiapkan organisasi untuk adopsi cloud yang sukses. Untuk informasi lebih lanjut, lihat situs web AWS CAF dan whitepaper AWS CAF.

AWS Kerangka Kualifikasi Beban Kerja (AWS WQF)

Alat yang mengevaluasi beban kerja migrasi database, merekomendasikan strategi migrasi, dan memberikan perkiraan kerja. AWS WQF disertakan dengan AWS Schema Conversion Tool ()AWS SCT. Ini menganalisis skema database dan objek kode, kode aplikasi, dependensi, dan karakteristik kinerja, dan memberikan laporan penilaian.

B

bot buruk

Bot yang dimaksudkan untuk mengganggu atau menyebabkan kerugian bagi individu atau organisasi.

BCP

Lihat perencanaan kontinuitas bisnis.

grafik perilaku

Pandangan interaktif yang terpadu tentang perilaku dan interaksi sumber daya dari waktu ke waktu. Anda dapat menggunakan grafik perilaku dengan Amazon Detective untuk memeriksa upaya logon yang gagal, panggilan API yang mencurigakan, dan tindakan serupa. Untuk informasi selengkapnya, lihat Data dalam grafik perilaku di dokumentasi Detektif.

sistem big-endian

Sistem yang menyimpan byte paling signifikan terlebih dahulu. Lihat juga endianness.

klasifikasi biner

Sebuah proses yang memprediksi hasil biner (salah satu dari dua kelas yang mungkin). Misalnya, model ML Anda mungkin perlu memprediksi masalah seperti “Apakah email ini spam atau bukan spam?” atau “Apakah produk ini buku atau mobil?”

filter mekar

Struktur data probabilistik dan efisien memori yang digunakan untuk menguji apakah suatu elemen adalah anggota dari suatu himpunan.

deployment biru/hijau

Strategi penyebaran tempat Anda membuat dua lingkungan yang terpisah namun identik. Anda menjalankan versi aplikasi saat ini di satu lingkungan (biru) dan versi aplikasi baru di lingkungan lain (hijau). Strategi ini membantu Anda dengan cepat memutar kembali dengan dampak minimal.

bot

Aplikasi perangkat lunak yang menjalankan tugas otomatis melalui internet dan mensimulasikan aktivitas atau interaksi manusia. Beberapa bot berguna atau bermanfaat, seperti perayap web yang mengindeks informasi di internet. Beberapa bot lain, yang dikenal sebagai bot buruk, dimaksudkan untuk mengganggu atau membahayakan individu atau organisasi.

botnet

Jaringan bot yang terinfeksi oleh malware dan berada di bawah kendali satu pihak, yang dikenal sebagai bot herder atau operator bot. Botnet adalah mekanisme paling terkenal untuk skala bot dan dampaknya.

cabang

Area berisi repositori kode. Cabang pertama yang dibuat dalam repositori adalah cabang utama. Anda dapat membuat cabang baru dari cabang yang ada, dan Anda kemudian dapat mengembangkan fitur atau memperbaiki bug di cabang baru. Cabang yang Anda buat untuk membangun fitur biasanya disebut sebagai cabang fitur. Saat fitur siap dirilis, Anda menggabungkan cabang fitur kembali ke cabang utama. Untuk informasi selengkapnya, lihat Tentang cabang (GitHub dokumentasi).

akses break-glass

Dalam keadaan luar biasa dan melalui proses yang disetujui, cara cepat bagi pengguna untuk mendapatkan akses ke Akun AWS yang biasanya tidak memiliki izin untuk mengaksesnya. Untuk informasi lebih lanjut, lihat indikator Implementasikan prosedur break-glass dalam panduan Well-Architected AWS .

strategi brownfield

Infrastruktur yang ada di lingkungan Anda. Saat mengadopsi strategi brownfield untuk arsitektur sistem, Anda merancang arsitektur di sekitar kendala sistem dan infrastruktur saat ini. Jika Anda memperluas infrastruktur yang ada, Anda dapat memadukan strategi brownfield dan greenfield.

cache penyangga

Area memori tempat data yang paling sering diakses disimpan.

kemampuan bisnis

Apa yang dilakukan bisnis untuk menghasilkan nilai (misalnya, penjualan, layanan pelanggan, atau pemasaran). Arsitektur layanan mikro dan keputusan pengembangan dapat didorong oleh kemampuan bisnis. Untuk informasi selengkapnya, lihat bagian Terorganisir di sekitar kemampuan bisnis dari Menjalankan layanan mikro kontainer di whitepaper. AWS

perencanaan kelangsungan bisnis (BCP)

Rencana yang membahas dampak potensial dari peristiwa yang mengganggu, seperti migrasi skala besar, pada operasi dan memungkinkan bisnis untuk melanjutkan operasi dengan cepat.

C

KAFE

Lihat Kerangka Adopsi AWS Cloud.

penyebaran kenari

Rilis versi yang lambat dan bertahap untuk pengguna akhir. Ketika Anda yakin, Anda menyebarkan versi baru dan mengganti versi saat ini secara keseluruhan.

CCoE

Lihat Cloud Center of Excellence.

CDC

Lihat mengubah pengambilan data.

ubah pengambilan data (CDC)

Proses melacak perubahan ke sumber data, seperti tabel database, dan merekam metadata tentang perubahan tersebut. Anda dapat menggunakan CDC untuk berbagai tujuan, seperti mengaudit atau mereplikasi perubahan dalam sistem target untuk mempertahankan sinkronisasi.

rekayasa kekacauan

Sengaja memperkenalkan kegagalan atau peristiwa yang mengganggu untuk menguji ketahanan sistem. Anda dapat menggunakan AWS Fault Injection Service (AWS FIS) untuk melakukan eksperimen yang menekankan AWS beban kerja Anda dan mengevaluasi responsnya.

CI/CD

Lihat integrasi berkelanjutan dan pengiriman berkelanjutan.

klasifikasi

Proses kategorisasi yang membantu menghasilkan prediksi. Model ML untuk masalah klasifikasi memprediksi nilai diskrit. Nilai diskrit selalu berbeda satu sama lain. Misalnya, model mungkin perlu mengevaluasi apakah ada mobil dalam gambar atau tidak.

Enkripsi sisi klien

Enkripsi data secara lokal, sebelum target Layanan AWS menerimanya.

Cloud Center of Excellence (CCoE)

Tim multi-disiplin yang mendorong upaya adopsi cloud di seluruh organisasi, termasuk mengembangkan praktik terbaik cloud, memobilisasi sumber daya, menetapkan jadwal migrasi, dan memimpin organisasi melalui transformasi skala besar. Untuk informasi selengkapnya, lihat posting CCoE di Blog Strategi AWS Cloud Perusahaan.

komputasi cloud

Teknologi cloud yang biasanya digunakan untuk penyimpanan data jarak jauh dan manajemen perangkat IoT. Cloud computing umumnya terhubung ke teknologi edge computing.

model operasi cloud

Dalam organisasi TI, model operasi yang digunakan untuk membangun, mematangkan, dan mengoptimalkan satu atau lebih lingkungan cloud. Untuk informasi selengkapnya, lihat Membangun Model Operasi Cloud Anda.

tahap adopsi cloud

Empat fase yang biasanya dilalui organisasi ketika mereka bermigrasi ke AWS Cloud:

- Proyek — Menjalankan beberapa proyek terkait cloud untuk bukti konsep dan tujuan pembelajaran
- Foundation — Melakukan investasi dasar untuk meningkatkan adopsi cloud Anda (misalnya, membuat landing zone, mendefinisikan CCoE, membuat model operasi)
- Migrasi — Migrasi aplikasi individual
- Re-invention — Mengoptimalkan produk dan layanan, dan berinovasi di cloud

Tahapan ini didefinisikan oleh Stephen Orban dalam posting blog The Journey Toward Cloud-First & the Stages of Adoption di blog Strategi Perusahaan. AWS Cloud Untuk informasi tentang bagaimana kaitannya dengan strategi AWS migrasi, lihat panduan kesiapan migrasi.

CMDB

Lihat database manajemen konfigurasi.

repositori kode

Lokasi di mana kode sumber dan aset lainnya, seperti dokumentasi, sampel, dan skrip, disimpan dan diperbarui melalui proses kontrol versi. Repositori cloud umum termasuk GitHub atau. AWS CodeCommit Setiap versi kode disebut cabang. Dalam struktur layanan mikro, setiap repositori

dikhususkan untuk satu bagian fungsionalitas. Pipa CI/CD tunggal dapat menggunakan beberapa repositori.

cache dingin

Cache buffer yang kosong, tidak terisi dengan baik, atau berisi data basi atau tidak relevan. Ini mempengaruhi kinerja karena instance database harus membaca dari memori utama atau disk, yang lebih lambat daripada membaca dari cache buffer.

data dingin

Data yang jarang diakses dan biasanya historis. Saat menanyakan jenis data ini, kueri lambat biasanya dapat diterima. Memindahkan data ini ke tingkat penyimpanan atau kelas yang berkinerja lebih rendah dan lebih murah dapat mengurangi biaya.

visi komputer (CV)

Bidang AI yang menggunakan pembelajaran mesin untuk menganalisis dan mengekstrak informasi dari format visual seperti gambar dan video digital. Misalnya, AWS Panorama menawarkan perangkat yang menambahkan CV ke jaringan kamera lokal, dan Amazon SageMaker menyediakan algoritme pemrosesan gambar untuk CV.

konfigurasi drift

Untuk beban kerja, konfigurasi berubah dari status yang diharapkan. Ini dapat menyebabkan beban kerja menjadi tidak patuh, dan biasanya bertahap dan tidak disengaja.

database manajemen konfigurasi (CMDB)

Repositori yang menyimpan dan mengelola informasi tentang database dan lingkungan TI, termasuk komponen perangkat keras dan perangkat lunak dan konfigurasinya. Anda biasanya menggunakan data dari CMDB dalam penemuan portofolio dan tahap analisis migrasi.

paket kesesuaian

Kumpulan AWS Config aturan dan tindakan remediasi yang dapat Anda kumpulkan untuk menyesuaikan kepatuhan dan pemeriksaan keamanan Anda. Anda dapat menerapkan paket kesesuaian sebagai entitas tunggal di Akun AWS dan Wilayah, atau di seluruh organisasi, dengan menggunakan templat YAMM. Untuk informasi selengkapnya, lihat Paket kesesuaian dalam dokumentasi. AWS Config

integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD)

Proses mengotomatiskan sumber, membangun, menguji, pementasan, dan tahap produksi dari proses rilis perangkat lunak. CI/CD umumnya digambarkan sebagai pipa. CI/CD dapat membantu

Anda mengotomatiskan proses, meningkatkan produktivitas, meningkatkan kualitas kode, dan memberikan lebih cepat. Untuk informasi lebih lanjut, lihat Manfaat pengiriman berkelanjutan. CD juga dapat berarti penerapan berkelanjutan. Untuk informasi selengkapnya, lihat Continuous Delivery vs Continuous Deployment.

CV

Lihat visi komputer.

D

data saat istirahat

Data yang stasioner di jaringan Anda, seperti data yang ada di penyimpanan.

klasifikasi data

Proses untuk mengidentifikasi dan mengkategorikan data dalam jaringan Anda berdasarkan kekritisan dan sensitivitasnya. Ini adalah komponen penting dari setiap strategi manajemen risiko keamanan siber karena membantu Anda menentukan perlindungan dan kontrol retensi yang tepat untuk data. Klasifikasi data adalah komponen pilar keamanan dalam AWS Well-Architected Framework. Untuk informasi selengkapnya, lihat Klasifikasi data.

penyimpangan data

Variasi yang berarti antara data produksi dan data yang digunakan untuk melatih model ML, atau perubahan yang berarti dalam data input dari waktu ke waktu. Penyimpangan data dapat mengurangi kualitas, akurasi, dan keadilan keseluruhan dalam prediksi model ML.

data dalam transit

Data yang aktif bergerak melalui jaringan Anda, seperti antara sumber daya jaringan.

jala data

Kerangka arsitektur yang menyediakan kepemilikan data terdistribusi dan terdesentralisasi dengan manajemen dan tata kelola terpusat.

minimalisasi data

Prinsip pengumpulan dan pemrosesan hanya data yang sangat diperlukan. Mempraktikkan minimalisasi data AWS Cloud dapat mengurangi risiko privasi, biaya, dan jejak karbon analitik Anda.

perimeter data

Satu set pagar pembatas pencegahan di AWS lingkungan Anda yang membantu memastikan bahwa hanya identitas tepercaya yang mengakses sumber daya tepercaya dari jaringan yang diharapkan. Untuk informasi selengkapnya, lihat Membangun perimeter data pada AWS.

prapemrosesan data

Untuk mengubah data mentah menjadi format yang mudah diuraikan oleh model ML Anda. Preprocessing data dapat berarti menghapus kolom atau baris tertentu dan menangani nilai yang hilang, tidak konsisten, atau duplikat.

asal data

Proses melacak asal dan riwayat data sepanjang siklus hidupnya, seperti bagaimana data dihasilkan, ditransmisikan, dan disimpan.

subjek data

Individu yang datanya dikumpulkan dan diproses.

gudang data

Sistem manajemen data yang mendukung intelijen bisnis, seperti analitik. Gudang data biasanya berisi sejumlah besar data historis, dan biasanya digunakan untuk kueri dan analisis.

bahasa definisi database (DDL)

Pernyataan atau perintah untuk membuat atau memodifikasi struktur tabel dan objek dalam database.

bahasa manipulasi basis data (DHTML)

Pernyataan atau perintah untuk memodifikasi (memasukkan, memperbarui, dan menghapus) informasi dalam database.

DDL

Lihat bahasa definisi database.

ansambel yang dalam

Untuk menggabungkan beberapa model pembelajaran mendalam untuk prediksi. Anda dapat menggunakan ansambel dalam untuk mendapatkan prediksi yang lebih akurat atau untuk memperkirakan ketidakpastian dalam prediksi.

pembelajaran mendalam

Subbidang ML yang menggunakan beberapa lapisan jaringan saraf tiruan untuk mengidentifikasi pemetaan antara data input dan variabel target yang diinginkan.

defense-in-depth

Pendekatan keamanan informasi di mana serangkaian mekanisme dan kontrol keamanan dilapisi dengan cermat di seluruh jaringan komputer untuk melindungi kerahasiaan, integritas, dan ketersediaan jaringan dan data di dalamnya. Saat Anda mengadopsi strategi ini AWS, Anda menambahkan beberapa kontrol pada lapisan AWS Organizations struktur yang berbeda untuk membantu mengamankan sumber daya. Misalnya, defense-in-depth pendekatan mungkin menggabungkan otentikasi multi-faktor, segmentasi jaringan, dan enkripsi.

administrator yang didelegasikan

Di AWS Organizations, layanan yang kompatibel dapat mendaftarkan akun AWS anggota untuk mengelola akun organisasi dan mengelola izin untuk layanan tersebut. Akun ini disebut administrator yang didelegasikan untuk layanan itu. Untuk informasi selengkapnya dan daftar layanan yang kompatibel, lihat Layanan yang berfungsi dengan AWS Organizations AWS Organizations dokumentasi.

deployment

Proses pembuatan aplikasi, fitur baru, atau perbaikan kode tersedia di lingkungan target. Deployment melibatkan penerapan perubahan dalam basis kode dan kemudian membangun dan menjalankan basis kode itu di lingkungan aplikasi.

lingkungan pengembangan

Lihat lingkungan.

kontrol detektif

Kontrol keamanan yang dirancang untuk mendeteksi, mencatat, dan memperingatkan setelah suatu peristiwa terjadi. Kontrol ini adalah garis pertahanan kedua, memperingatkan Anda tentang peristiwa keamanan yang melewati kontrol pencegahan di tempat. Untuk informasi selengkapnya, lihat Kontrol Detektif dalam Menerapkan kontrol keamanan pada. AWS

pemetaan aliran nilai pengembangan (DVSM)

Sebuah proses yang digunakan untuk mengidentifikasi dan memprioritaskan kendala yang mempengaruhi kecepatan dan kualitas dalam siklus hidup pengembangan perangkat lunak. DVSM memperluas proses pemetaan aliran nilai yang awalnya dirancang untuk praktik

manufaktur ramping. Ini berfokus pada langkah-langkah dan tim yang diperlukan untuk menciptakan dan memindahkan nilai melalui proses pengembangan perangkat lunak.

kembar digital

Representasi virtual dari sistem dunia nyata, seperti bangunan, pabrik, peralatan industri, atau jalur produksi. Kembar digital mendukung pemeliharaan prediktif, pemantauan jarak jauh, dan optimalisasi produksi.

tabel dimensi

Dalam skema bintang, tabel yang lebih kecil yang berisi atribut data tentang data kuantitatif dalam tabel fakta. Atribut tabel dimensi biasanya bidang teks atau angka diskrit yang berperilaku seperti teks. Atribut ini biasanya digunakan untuk pembatasan kueri, pemfilteran, dan pelabelan set hasil.

musibah

Peristiwa yang mencegah beban kerja atau sistem memenuhi tujuan bisnisnya di lokasi utama yang digunakan. Peristiwa ini dapat berupa bencana alam, kegagalan teknis, atau akibat dari tindakan manusia, seperti kesalahan konfigurasi yang tidak disengaja atau serangan malware.

pemulihan bencana (DR)

Strategi dan proses yang Anda gunakan untuk meminimalkan downtime dan kehilangan data yang disebabkan oleh bencana. Untuk informasi selengkapnya, lihat Disaster Recovery of Workloads on AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML~

Lihat bahasa manipulasi database.

desain berbasis domain

Pendekatan untuk mengembangkan sistem perangkat lunak yang kompleks dengan menghubungkan komponennya ke domain yang berkembang, atau tujuan bisnis inti, yang dilayani oleh setiap komponen. Konsep ini diperkenalkan oleh Eric Evans dalam bukunya, Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston: Addison-Wesley Professional, 2003). Untuk informasi tentang cara menggunakan desain berbasis domain dengan pola gambar pencekik, lihat Memodernisasi layanan web Microsoft ASP.NET (ASMX) lama secara bertahap menggunakan container dan Amazon API Gateway.

DR

Lihat pemulihan bencana.

deteksi drift

Melacak penyimpangan dari konfigurasi dasar. Misalnya, Anda dapat menggunakan AWS CloudFormation untuk mendeteksi penyimpangan dalam sumber daya sistem, atau Anda dapat menggunakannya AWS Control Tower untuk mendeteksi perubahan di landing zone yang mungkin memengaruhi kepatuhan terhadap persyaratan tata kelola.

DVSM

Lihat pemetaan aliran nilai pengembangan.

E

EDA

Lihat analisis data eksplorasi.

komputasi tepi

Teknologi yang meningkatkan daya komputasi untuk perangkat pintar di tepi jaringan IoT. Jika dibandingkan dengan komputasi awan, komputasi tepi dapat mengurangi latensi komunikasi dan meningkatkan waktu respons.

enkripsi

Proses komputasi yang mengubah data plaintext, yang dapat dibaca manusia, menjadi ciphertext.

kunci enkripsi

String kriptografi dari bit acak yang dihasilkan oleh algoritma enkripsi. Panjang kunci dapat bervariasi, dan setiap kunci dirancang agar tidak dapat diprediksi dan unik.

endianness

Urutan byte disimpan dalam memori komputer. Sistem big-endian menyimpan byte paling signifikan terlebih dahulu. Sistem little-endian menyimpan byte paling tidak signifikan terlebih dahulu.

titik akhir

Lihat titik akhir layanan.

layanan endpoint

Layanan yang dapat Anda host di cloud pribadi virtual (VPC) untuk dibagikan dengan pengguna lain. Anda dapat membuat layanan endpoint dengan AWS PrivateLink dan memberikan izin kepada prinsipal lain Akun AWS atau ke AWS Identity and Access Management (IAM). Akun atau prinsipal ini dapat terhubung ke layanan endpoint Anda secara pribadi dengan membuat titik akhir VPC antarmuka. Untuk informasi selengkapnya, lihat Membuat layanan titik akhir di dokumentasi Amazon Virtual Private Cloud (Amazon VPC).

perencanaan sumber daya perusahaan (ERP)

Sistem yang mengotomatiskan dan mengelola proses bisnis utama (seperti akuntansi, MES, dan manajemen proyek) untuk suatu perusahaan.

enkripsi amplop

Proses mengenkripsi kunci enkripsi dengan kunci enkripsi lain. Untuk informasi selengkapnya, lihat Enkripsi amplop dalam dokumentasi AWS Key Management Service (AWS KMS).

lingkungan

Sebuah contoh dari aplikasi yang sedang berjalan. Berikut ini adalah jenis lingkungan yang umum dalam komputasi awan:

- Development Environment — Sebuah contoh dari aplikasi yang berjalan yang hanya tersedia untuk tim inti yang bertanggung jawab untuk memelihara aplikasi. Lingkungan pengembangan digunakan untuk menguji perubahan sebelum mempromosikannya ke lingkungan atas. Jenis lingkungan ini kadang-kadang disebut sebagai lingkungan pengujian.
- lingkungan yang lebih rendah — Semua lingkungan pengembangan untuk aplikasi, seperti yang digunakan untuk build awal dan pengujian.
- lingkungan produksi — Sebuah contoh dari aplikasi yang berjalan yang pengguna akhir dapat mengakses. Dalam pipa CI/CD, lingkungan produksi adalah lingkungan penyebaran terakhir.
- lingkungan atas — Semua lingkungan yang dapat diakses oleh pengguna selain tim pengembangan inti. Ini dapat mencakup lingkungan produksi, lingkungan praproduksi, dan lingkungan untuk pengujian penerimaan pengguna.

epik

Dalam metodologi tangkas, kategori fungsional yang membantu mengatur dan memprioritaskan pekerjaan Anda. Epik memberikan deskripsi tingkat tinggi tentang persyaratan dan tugas

implementasi. Misalnya, epos keamanan AWS CAF mencakup manajemen identitas dan akses, kontrol detektif, keamanan infrastruktur, perlindungan data, dan respons insiden. Untuk informasi selengkapnya tentang epos dalam strategi AWS migrasi, lihat panduan implementasi program.

ERP

Lihat perencanaan sumber daya perusahaan.

analisis data eksplorasi (EDA)

Proses menganalisis dataset untuk memahami karakteristik utamanya. Anda mengumpulkan atau mengumpulkan data dan kemudian melakukan penyelidikan awal untuk menemukan pola, mendeteksi anomali, dan memeriksa asumsi. EDA dilakukan dengan menghitung statistik ringkasan dan membuat visualisasi data.

F

tabel fakta

Tabel tengah dalam skema bintang. Ini menyimpan data kuantitatif tentang operasi bisnis. Biasanya, tabel fakta berisi dua jenis kolom: kolom yang berisi ukuran dan yang berisi kunci asing ke tabel dimensi.

gagal cepat

Filosofi yang menggunakan pengujian yang sering dan bertahap untuk mengurangi siklus hidup pengembangan. Ini adalah bagian penting dari pendekatan tangkas.

batas isolasi kesalahan

Dalam AWS Cloud, batas seperti Availability Zone, Wilayah AWS, control plane, atau data plane yang membatasi efek kegagalan dan membantu meningkatkan ketahanan beban kerja. Untuk informasi selengkapnya, lihat Batas Isolasi AWS Kesalahan.

cabang fitur

Lihat cabang.

fitur

Data input yang Anda gunakan untuk membuat prediksi. Misalnya, dalam konteks manufaktur, fitur bisa berupa gambar yang diambil secara berkala dari lini manufaktur.

pentingnya fitur

Seberapa signifikan fitur untuk prediksi model. Ini biasanya dinyatakan sebagai skor numerik yang dapat dihitung melalui berbagai teknik, seperti Shapley Additive Explanations (SHAP) dan gradien terintegrasi. Untuk informasi lebih lanjut, lihat Interpretabilitas model pembelajaran mesin dengan:.AWS

transformasi fitur

Untuk mengoptimalkan data untuk proses ML, termasuk memperkaya data dengan sumber tambahan, menskalakan nilai, atau mengekstrak beberapa set informasi dari satu bidang data. Hal ini memungkinkan model ML untuk mendapatkan keuntungan dari data. Misalnya, jika Anda memecah tanggal “2021-05-27 00:15:37" menjadi “2021", “Mei”, “Kamis”, dan “15", Anda dapat membantu algoritme pembelajaran mempelajari pola bernuansa yang terkait dengan komponen data yang berbeda.

FGAC

Lihat kontrol akses berbutir halus.

kontrol akses berbutir halus (FGAC)

Penggunaan beberapa kondisi untuk mengizinkan atau menolak permintaan akses.

migrasi flash-cut

Metode migrasi database yang menggunakan replikasi data berkelanjutan melalui pengambilan data perubahan untuk memigrasikan data dalam waktu sesingkat mungkin, alih-alih menggunakan pendekatan bertahap. Tujuannya adalah untuk menjaga downtime seminimal mungkin.

G

pemblokiran geografis

Lihat pembatasan geografis.

pembatasan geografis (pemblokiran geografis)

Di Amazon CloudFront, opsi untuk mencegah pengguna di negara tertentu mengakses distribusi konten. Anda dapat menggunakan daftar izinkan atau daftar blokir untuk menentukan negara yang disetujui dan dilarang. Untuk informasi selengkapnya, lihat Membatasi distribusi geografis konten Anda dalam dokumentasi. CloudFront

Alur kerja Gitflow

Pendekatan di mana lingkungan bawah dan atas menggunakan cabang yang berbeda dalam repositori kode sumber. Alur kerja Gitflow dianggap warisan, dan alur kerja berbasis batang adalah pendekatan modern yang lebih disukai.

strategi greenfield

Tidak adanya infrastruktur yang ada di lingkungan baru. Saat mengadopsi strategi greenfield untuk arsitektur sistem, Anda dapat memilih semua teknologi baru tanpa batasan kompatibilitas dengan infrastruktur yang ada, juga dikenal sebagai brownfield. Jika Anda memperluas infrastruktur yang ada, Anda dapat memadukan strategi brownfield dan greenfield.

pagar pembatas

Aturan tingkat tinggi yang membantu mengatur sumber daya, kebijakan, dan kepatuhan di seluruh unit organisasi (OU). Pagar pembatas preventif menegakkan kebijakan untuk memastikan keselarasan dengan standar kepatuhan. Mereka diimplementasikan dengan menggunakan kebijakan kontrol layanan dan batas izin IAM. Detective guardrails mendeteksi pelanggaran kebijakan dan masalah kepatuhan, dan menghasilkan peringatan untuk remediasi. Mereka diimplementasikan dengan menggunakan AWS Config, AWS Security Hub, Amazon GuardDuty AWS Trusted Advisor, Amazon Inspector, dan pemeriksaan khusus AWS Lambda .

H

HA

Lihat ketersediaan tinggi.

migrasi database heterogen

Memigrasi database sumber Anda ke database target yang menggunakan mesin database yang berbeda (misalnya, Oracle ke Amazon Aurora). Migrasi heterogen biasanya merupakan bagian dari upaya arsitektur ulang, dan mengubah skema dapat menjadi tugas yang kompleks. AWS menyediakan AWS SCT yang membantu dengan konversi skema.

ketersediaan tinggi (HA)

Kemampuan beban kerja untuk beroperasi terus menerus, tanpa intervensi, jika terjadi tantangan atau bencana. Sistem HA dirancang untuk gagal secara otomatis, secara konsisten memberikan kinerja berkualitas tinggi, dan menangani beban dan kegagalan yang berbeda dengan dampak kinerja minimal.

modernisasi sejarawan

Pendekatan yang digunakan untuk memodernisasi dan meningkatkan sistem teknologi operasional (OT) untuk melayani kebutuhan industri manufaktur dengan lebih baik. Sejarawan adalah jenis database yang digunakan untuk mengumpulkan dan menyimpan data dari berbagai sumber di pabrik.

migrasi database homogen

Memigrasi database sumber Anda ke database target yang berbagi mesin database yang sama (misalnya, Microsoft SQL Server ke Amazon RDS for SQL Server). Migrasi homogen biasanya merupakan bagian dari upaya rehosting atau replatforming. Anda dapat menggunakan utilitas database asli untuk memigrasi skema.

data panas

Data yang sering diakses, seperti data real-time atau data translasi terbaru. Data ini biasanya memerlukan tingkat atau kelas penyimpanan berkinerja tinggi untuk memberikan respons kueri yang cepat.

perbaikan terbaru

Perbaikan mendesak untuk masalah kritis dalam lingkungan produksi. Karena urgensinya, perbaikan terbaru biasanya dibuat di luar alur kerja DevOps rilis biasa.

periode hypercare

Segera setelah cutover, periode waktu ketika tim migrasi mengelola dan memantau aplikasi yang dimigrasi di cloud untuk mengatasi masalah apa pun. Biasanya, periode ini panjangnya 1-4 hari. Pada akhir periode hypercare, tim migrasi biasanya mentransfer tanggung jawab untuk aplikasi ke tim operasi cloud.

I

IAc

Lihat infrastruktur sebagai kode.

kebijakan berbasis identitas

Kebijakan yang dilampirkan pada satu atau beberapa prinsip IAM yang mendefinisikan izin mereka dalam lingkungan. AWS Cloud

aplikasi idle

Aplikasi yang memiliki penggunaan CPU dan memori rata-rata antara 5 dan 20 persen selama periode 90 hari. Dalam proyek migrasi, adalah umum untuk menghentikan aplikasi ini atau mempertahankannya di tempat.

IIoT

Lihat Internet of Things industri.

infrastruktur yang tidak dapat diubah

Model yang menyebarkan infrastruktur baru untuk beban kerja produksi alih-alih memperbarui, menambal, atau memodifikasi infrastruktur yang ada. Infrastruktur yang tidak dapat diubah secara inheren lebih konsisten, andal, dan dapat diprediksi daripada infrastruktur yang dapat berubah. Untuk informasi selengkapnya, lihat praktik terbaik Deploy using immutable infrastructure di AWS Well-Architected Framework.

masuk (masuknya) VPC

Dalam arsitektur AWS multi-akun, VPC yang menerima, memeriksa, dan merutekan koneksi jaringan dari luar aplikasi. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan Anda dengan VPC masuk, keluar, dan inspeksi untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

migrasi inkremental

Strategi cutover di mana Anda memigrasikan aplikasi Anda dalam bagian-bagian kecil alih-alih melakukan satu cutover penuh. Misalnya, Anda mungkin hanya memindahkan beberapa layanan mikro atau pengguna ke sistem baru pada awalnya. Setelah Anda memverifikasi bahwa semuanya berfungsi dengan baik, Anda dapat secara bertahap memindahkan layanan mikro atau pengguna tambahan hingga Anda dapat menonaktifkan sistem lama Anda. Strategi ini mengurangi risiko yang terkait dengan migrasi besar.

Industri 4.0

Sebuah istilah yang diperkenalkan oleh Klaus Schwab pada tahun 2016 untuk merujuk pada modernisasi proses manufaktur melalui kemajuan dalam konektivitas, data real-time, otomatisasi, analitik, dan AI/ML.

infrastruktur

Semua sumber daya dan aset yang terkandung dalam lingkungan aplikasi.

infrastruktur sebagai kode (IAc)

Proses penyediaan dan pengelolaan infrastruktur aplikasi melalui satu set file konfigurasi. IAc dirancang untuk membantu Anda memusatkan manajemen infrastruktur, menstandarisasi sumber daya, dan menskalakan dengan cepat sehingga lingkungan baru dapat diulang, andal, dan konsisten.

Internet of Things industri (IIoT)

Penggunaan sensor dan perangkat yang terhubung ke internet di sektor industri, seperti manufaktur, energi, otomotif, perawatan kesehatan, ilmu kehidupan, dan pertanian. Untuk informasi selengkapnya, lihat Membangun strategi transformasi digital Internet of Things (IIoT) industri.

inspeksi VPC

Dalam arsitektur AWS multi-akun, VPC terpusat yang mengelola inspeksi lalu lintas jaringan antara VPC (dalam hal yang sama atau berbeda Wilayah AWS), internet, dan jaringan lokal. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan Anda dengan VPC masuk, keluar, dan inspeksi untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

Internet of Things (IoT)

Jaringan objek fisik yang terhubung dengan sensor atau prosesor tertanam yang berkomunikasi dengan perangkat dan sistem lain melalui internet atau melalui jaringan komunikasi lokal. Untuk informasi selengkapnya, lihat Apa itu IoT?

interpretabilitas

Karakteristik model pembelajaran mesin yang menggambarkan sejauh mana manusia dapat memahami bagaimana prediksi model bergantung pada inputnya. Untuk informasi selengkapnya, lihat Interpretabilitas model pembelajaran mesin dengan AWS.

IoT

Lihat Internet of Things.

Perpustakaan informasi TI (ITIL)

Serangkaian praktik terbaik untuk memberikan layanan TI dan menyelaraskan layanan ini dengan persyaratan bisnis. ITIL menyediakan dasar untuk ITSM.

Manajemen layanan TI (ITSM)

Kegiatan yang terkait dengan merancang, menerapkan, mengelola, dan mendukung layanan TI untuk suatu organisasi. Untuk informasi tentang mengintegrasikan operasi cloud dengan alat ITSM, lihat panduan integrasi operasi.

ITIL

Lihat perpustakaan informasi TI.

ITSM

Lihat manajemen layanan TI.

L

kontrol akses berbasis label (LBAC)

Implementasi kontrol akses wajib (MAC) di mana pengguna dan data itu sendiri masing-masing secara eksplisit diberi nilai label keamanan. Persimpangan antara label keamanan pengguna dan label keamanan data menentukan baris dan kolom mana yang dapat dilihat oleh pengguna.

landing zone

Landing zone adalah AWS lingkungan multi-akun yang dirancang dengan baik yang dapat diskalakan dan aman. Ini adalah titik awal dari mana organisasi Anda dapat dengan cepat meluncurkan dan menyebarkan beban kerja dan aplikasi dengan percaya diri dalam lingkungan keamanan dan infrastruktur mereka. Untuk informasi selengkapnya tentang zona pendaratan, lihat Menyiapkan lingkungan multi-akun AWS yang aman dan dapat diskalakan.

migrasi besar

Migrasi 300 atau lebih server.

LBAC

Lihat kontrol akses berbasis label.

hak istimewa paling sedikit

Praktik keamanan terbaik untuk memberikan izin minimum yang diperlukan untuk melakukan tugas. Untuk informasi selengkapnya, lihat Menerapkan izin hak istimewa terkecil dalam dokumentasi IAM.

angkat dan geser

Lihat 7 Rs.

sistem endian kecil

Sebuah sistem yang menyimpan byte paling tidak signifikan terlebih dahulu. Lihat juga endianness.

lingkungan yang lebih rendah

Lihat lingkungan.

M

pembelajaran mesin (ML)

Jenis kecerdasan buatan yang menggunakan algoritma dan teknik untuk pengenalan pola dan pembelajaran. ML menganalisis dan belajar dari data yang direkam, seperti data Internet of Things (IoT), untuk menghasilkan model statistik berdasarkan pola. Untuk informasi selengkapnya, lihat Machine Learning.

cabang utama

Lihat cabang.

malware

Perangkat lunak yang dirancang untuk membahayakan keamanan atau privasi komputer. Malware dapat mengganggu sistem komputer, membocorkan informasi sensitif, atau mendapatkan akses yang tidak sah. Contoh malware termasuk virus, worm, ransomware, Trojan horse, spyware, dan keyloggers.

layanan terkelola

Layanan AWS yang AWS mengoperasikan lapisan infrastruktur, sistem operasi, dan platform, dan Anda mengakses titik akhir untuk menyimpan dan mengambil data. Amazon Simple Storage Service (Amazon S3) dan Amazon DynamoDB adalah contoh layanan terkelola. Ini juga dikenal sebagai layanan abstrak.

sistem eksekusi manufaktur (MES)

Sistem perangkat lunak untuk melacak, memantau, mendokumentasikan, dan mengendalikan proses produksi yang mengubah bahan baku menjadi produk jadi di lantai toko.

PETA

Lihat Program Percepatan Migrasi.

mekanisme

Proses lengkap di mana Anda membuat alat, mendorong adopsi alat, dan kemudian memeriksa hasilnya untuk melakukan penyesuaian. Mekanisme adalah siklus yang memperkuat dan meningkatkan dirinya sendiri saat beroperasi. Untuk informasi lebih lanjut, lihat Membangun mekanisme di AWS Well-Architected Framework.

akun anggota

Semua Akun AWS selain akun manajemen yang merupakan bagian dari organisasi di AWS Organizations. Akun dapat menjadi anggota dari hanya satu organisasi pada suatu waktu.

MES

Lihat sistem eksekusi manufaktur.

Transportasi Telemetri Antrian Pesan (MQTT)

Protokol komunikasi ringan machine-to-machine (M2M), berdasarkan pola terbitkan/berlangganan, untuk perangkat IoT yang dibatasi sumber daya.

layanan mikro

Layanan kecil dan independen yang berkomunikasi melalui API yang terdefinisi dengan baik dan biasanya dimiliki oleh tim kecil yang mandiri. Misalnya, sistem asuransi mungkin mencakup layanan mikro yang memetakan kemampuan bisnis, seperti penjualan atau pemasaran, atau subdomain, seperti pembelian, klaim, atau analitik. Manfaat layanan mikro termasuk kelincahan, penskalaan yang fleksibel, penyebaran yang mudah, kode yang dapat digunakan kembali, dan ketahanan. Untuk informasi selengkapnya, lihat Mengintegrasikan layanan mikro dengan menggunakan layanan tanpa AWS server.

arsitektur microservices

Pendekatan untuk membangun aplikasi dengan komponen independen yang menjalankan setiap proses aplikasi sebagai layanan mikro. Layanan mikro ini berkomunikasi melalui antarmuka yang terdefinisi dengan baik dengan menggunakan API ringan. Setiap layanan mikro dalam arsitektur ini dapat diperbarui, digunakan, dan diskalakan untuk memenuhi permintaan fungsi tertentu dari suatu aplikasi. Untuk informasi selengkapnya, lihat Menerapkan layanan mikro di AWS.

Program Percepatan Migrasi (MAP)

AWS Program yang menyediakan dukungan konsultasi, pelatihan, dan layanan untuk membantu organisasi membangun fondasi operasional yang kuat untuk pindah ke cloud, dan untuk membantu mengimbangi biaya awal migrasi. MAP mencakup metodologi migrasi untuk mengeksekusi migrasi lama dengan cara metodis dan seperangkat alat untuk mengotomatisasi dan mempercepat skenario migrasi umum.

migrasi dalam skala

Proses memindahkan sebagian besar portofolio aplikasi ke cloud dalam gelombang, dengan lebih banyak aplikasi bergerak pada tingkat yang lebih cepat di setiap gelombang. Fase ini menggunakan praktik terbaik dan pelajaran yang dipetik dari fase sebelumnya untuk mengimplementasikan pabrik migrasi tim, alat, dan proses untuk merampingkan migrasi beban kerja melalui otomatisasi dan pengiriman tangkas. Ini adalah fase ketiga dari strategi AWS migrasi.

pabrik migrasi

Tim lintas fungsi yang merampingkan migrasi beban kerja melalui pendekatan otomatis dan gesit. Tim pabrik migrasi biasanya mencakup operasi, analis dan pemilik bisnis, insinyur migrasi, pengembang, dan DevOps profesional yang bekerja di sprint. Antara 20 dan 50 persen portofolio aplikasi perusahaan terdiri dari pola berulang yang dapat dioptimalkan dengan pendekatan pabrik. Untuk informasi selengkapnya, lihat diskusi tentang pabrik migrasi dan panduan Pabrik Migrasi Cloud di kumpulan konten ini.

metadata migrasi

Informasi tentang aplikasi dan server yang diperlukan untuk menyelesaikan migrasi. Setiap pola migrasi memerlukan satu set metadata migrasi yang berbeda. Contoh metadata migrasi termasuk subnet target, grup keamanan, dan akun. AWS

pola migrasi

Tugas migrasi berulang yang merinci strategi migrasi, tujuan migrasi, dan aplikasi atau layanan migrasi yang digunakan. Contoh: Rehost migrasi ke Amazon EC2 AWS dengan Layanan Migrasi Aplikasi.

Penilaian Portofolio Migrasi (MPA)

Alat online yang menyediakan informasi untuk memvalidasi kasus bisnis untuk bermigrasi ke. AWS Cloud MPA menyediakan penilaian portofolio terperinci (ukuran kanan server, harga,

perbandingan TCO, analisis biaya migrasi) serta perencanaan migrasi (analisis data aplikasi dan pengumpulan data, pengelompokan aplikasi, prioritas migrasi, dan perencanaan gelombang). Alat MPA (memerlukan login) tersedia gratis untuk semua AWS konsultan dan konsultan APN Partner.

Penilaian Kesiapan Migrasi (MRA)

Proses mendapatkan wawasan tentang status kesiapan cloud organisasi, mengidentifikasi kekuatan dan kelemahan, dan membangun rencana aksi untuk menutup kesenjangan yang diidentifikasi, menggunakan CAF. AWS Untuk informasi selengkapnya, lihat panduan kesiapan migrasi. MRA adalah tahap pertama dari strategi AWS migrasi.

strategi migrasi

Pendekatan yang digunakan untuk memigrasikan beban kerja ke file. AWS Cloud Untuk informasi lebih lanjut, lihat entri 7 Rs di glosarium ini dan lihat Memobilisasi organisasi Anda untuk mempercepat migrasi skala besar.

ML

Lihat pembelajaran mesin.

modernisasi

Mengubah aplikasi usang (warisan atau monolitik) dan infrastrukturnya menjadi sistem yang gesit, elastis, dan sangat tersedia di cloud untuk mengurangi biaya, mendapatkan efisiensi, dan memanfaatkan inovasi. Untuk informasi selengkapnya, lihat Strategi untuk memodernisasi aplikasi di. AWS Cloud

penilaian kesiapan modernisasi

Evaluasi yang membantu menentukan kesiapan modernisasi aplikasi organisasi; mengidentifikasi manfaat, risiko, dan dependensi; dan menentukan seberapa baik organisasi dapat mendukung keadaan masa depan aplikasi tersebut. Hasil penilaian adalah cetak biru arsitektur target, peta jalan yang merinci fase pengembangan dan tonggak untuk proses modernisasi, dan rencana aksi untuk mengatasi kesenjangan yang diidentifikasi. Untuk informasi lebih lanjut, lihat Mengevaluasi kesiapan modernisasi untuk aplikasi di. AWS Cloud

aplikasi monolitik (monolit)

Aplikasi yang berjalan sebagai layanan tunggal dengan proses yang digabungkan secara ketat. Aplikasi monolitik memiliki beberapa kelemahan. Jika satu fitur aplikasi mengalami lonjakan permintaan, seluruh arsitektur harus diskalakan. Menambahkan atau meningkatkan fitur aplikasi monolitik juga menjadi lebih kompleks ketika basis kode tumbuh. Untuk mengatasi masalah ini,

Anda dapat menggunakan arsitektur microservices. Untuk informasi lebih lanjut, lihat Mengurai monolit menjadi layanan mikro.

MPA

Lihat Penilaian Portofolio Migrasi.

MQTT

Lihat Transportasi Telemetri Antrian Pesan.

klasifikasi multiclass

Sebuah proses yang membantu menghasilkan prediksi untuk beberapa kelas (memprediksi satu dari lebih dari dua hasil). Misalnya, model ML mungkin bertanya "Apakah produk ini buku, mobil, atau telepon?" atau "Kategori produk mana yang paling menarik bagi pelanggan ini?"

infrastruktur yang bisa berubah

Model yang memperbarui dan memodifikasi infrastruktur yang ada untuk beban kerja produksi. Untuk meningkatkan konsistensi, keandalan, dan prediktabilitas, AWS Well-Architected Framework merekomendasikan penggunaan infrastruktur yang tidak dapat diubah sebagai praktik terbaik.

O

OAC

Lihat kontrol akses asal.

OAI

Lihat identitas akses asal.

OCM

Lihat manajemen perubahan organisasi.

migrasi offline

Metode migrasi di mana beban kerja sumber diturunkan selama proses migrasi. Metode ini melibatkan waktu henti yang diperpanjang dan biasanya digunakan untuk beban kerja kecil dan tidak kritis.

OI

Lihat integrasi operasi.

OLA

Lihat perjanjian tingkat operasional.

migrasi online

Metode migrasi di mana beban kerja sumber disalin ke sistem target tanpa diambil offline. Aplikasi yang terhubung ke beban kerja dapat terus berfungsi selama migrasi. Metode ini melibatkan waktu henti nol hingga minimal dan biasanya digunakan untuk beban kerja produksi yang kritis.

OPC-UA

Lihat Komunikasi Proses Terbuka - Arsitektur Terpadu.

Komunikasi Proses Terbuka - Arsitektur Terpadu (OPC-UA)

Protokol komunikasi machine-to-machine (M2M) untuk otomasi industri. OPC-UA menyediakan standar interoperabilitas dengan enkripsi data, otentikasi, dan skema otorisasi.

perjanjian tingkat operasional (OLA)

Perjanjian yang menjelaskan apa yang dijanjikan kelompok TI fungsional untuk diberikan satu sama lain, untuk mendukung perjanjian tingkat layanan (SLA).

Tinjauan Kesiapan Operasional (ORR)

Daftar pertanyaan dan praktik terbaik terkait yang membantu Anda memahami, mengevaluasi, mencegah, atau mengurangi ruang lingkup insiden dan kemungkinan kegagalan. Untuk informasi lebih lanjut, lihat Ulasan Kesiapan Operasional (ORR) dalam Kerangka Kerja Well-Architected AWS .

teknologi operasional (OT)

Sistem perangkat keras dan perangkat lunak yang bekerja dengan lingkungan fisik untuk mengendalikan operasi industri, peralatan, dan infrastruktur. Di bidang manufaktur, integrasi sistem OT dan teknologi informasi (TI) adalah fokus utama untuk transformasi Industri 4.0.

integrasi operasi (OI)

Proses modernisasi operasi di cloud, yang melibatkan perencanaan kesiapan, otomatisasi, dan integrasi. Untuk informasi selengkapnya, lihat panduan integrasi operasi.

jejak organisasi

Jejak yang dibuat oleh AWS CloudTrail itu mencatat semua peristiwa untuk semua Akun AWS dalam organisasi di AWS Organizations. Jejak ini dibuat di setiap Akun AWS bagian organisasi dan melacak aktivitas di setiap akun. Untuk informasi selengkapnya, lihat Membuat jejak untuk organisasi dalam CloudTrail dokumentasi.

manajemen perubahan organisasi (OCM)

Kerangka kerja untuk mengelola transformasi bisnis utama yang mengganggu dari perspektif orang, budaya, dan kepemimpinan. OCM membantu organisasi mempersiapkan, dan transisi ke, sistem dan strategi baru dengan mempercepat adopsi perubahan, mengatasi masalah transisi, dan mendorong perubahan budaya dan organisasi. Dalam strategi AWS migrasi, kerangka kerja ini disebut percepatan orang, karena kecepatan perubahan yang diperlukan dalam proyek adopsi cloud. Untuk informasi lebih lanjut, lihat panduan OCM.

kontrol akses asal (OAC)

Di CloudFront, opsi yang disempurnakan untuk membatasi akses untuk mengamankan konten Amazon Simple Storage Service (Amazon S3) Anda. OAC mendukung semua bucket S3 di semua Wilayah AWS, enkripsi sisi server dengan AWS KMS (SSE-KMS), dan dinamis dan permintaan ke bucket S3. `PUT` `DELETE`

identitas akses asal (OAI)

Di CloudFront, opsi untuk membatasi akses untuk mengamankan konten Amazon S3 Anda. Saat Anda menggunakan OAI, CloudFront buat prinsipal yang dapat diautentikasi oleh Amazon S3. Prinsipal yang diautentikasi dapat mengakses konten dalam bucket S3 hanya melalui distribusi tertentu. CloudFront Lihat juga OAC, yang menyediakan kontrol akses yang lebih terperinci dan ditingkatkan.

ORR

Lihat tinjauan kesiapan operasional.

OT

Lihat teknologi operasional.

keluar (jalan keluar) VPC

Dalam arsitektur AWS multi-akun, VPC yang menangani koneksi jaringan yang dimulai dari dalam aplikasi. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan

Anda dengan VPC masuk, keluar, dan inspeksi untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

P

batas izin

Kebijakan manajemen IAM yang dilampirkan pada prinsipal IAM untuk menetapkan izin maksimum yang dapat dimiliki pengguna atau peran. Untuk informasi selengkapnya, lihat Batas izin dalam dokumentasi IAM.

Informasi Identifikasi Pribadi (PII)

Informasi yang, jika dilihat secara langsung atau dipasangkan dengan data terkait lainnya, dapat digunakan untuk menyimpulkan identitas individu secara wajar. Contoh PII termasuk nama, alamat, dan informasi kontak.

PII

Lihat informasi yang dapat diidentifikasi secara pribadi.

buku pedoman

Serangkaian langkah yang telah ditentukan sebelumnya yang menangkap pekerjaan yang terkait dengan migrasi, seperti mengirimkan fungsi operasi inti di cloud. Buku pedoman dapat berupa skrip, runbook otomatis, atau ringkasan proses atau langkah-langkah yang diperlukan untuk mengoperasikan lingkungan modern Anda.

PLC

Lihat pengontrol logika yang dapat diprogram.

PLM

Lihat manajemen siklus hidup produk.

kebijakan

Objek yang dapat menentukan izin (lihat kebijakan berbasis identitas), menentukan kondisi akses (lihat kebijakanberbasis sumber daya), atau menentukan izin maksimum untuk semua akun dalam organisasi di (lihat kebijakan kontrol layanan). AWS Organizations

persistensi poliglot

Secara independen memilih teknologi penyimpanan data microservice berdasarkan pola akses data dan persyaratan lainnya. Jika layanan mikro Anda memiliki teknologi penyimpanan data yang sama, mereka dapat menghadapi tantangan implementasi atau mengalami kinerja yang buruk. Layanan mikro lebih mudah diimplementasikan dan mencapai kinerja dan skalabilitas yang lebih baik jika mereka menggunakan penyimpanan data yang paling sesuai dengan kebutuhan mereka. Untuk informasi selengkapnya, lihat Mengaktifkan persistensi data di layanan mikro.

penilaian portofolio

Proses menemukan, menganalisis, dan memprioritaskan portofolio aplikasi untuk merencanakan migrasi. Untuk informasi selengkapnya, lihat Mengevaluasi kesiapan migrasi.

predikat

Kondisi kueri yang mengembalikan `true` atau`false`, biasanya terletak di `WHERE` klausa.

predikat pushdown

Teknik pengoptimalan kueri database yang menyaring data dalam kueri sebelum transfer. Ini mengurangi jumlah data yang harus diambil dan diproses dari database relasional, dan meningkatkan kinerja kueri.

kontrol preventif

Kontrol keamanan yang dirancang untuk mencegah suatu peristiwa terjadi. Kontrol ini adalah garis pertahanan pertama untuk membantu mencegah akses tidak sah atau perubahan yang tidak diinginkan ke jaringan Anda. Untuk informasi selengkapnya, lihat Kontrol pencegahan dalam Menerapkan kontrol keamanan pada. AWS

principal

Entitas AWS yang dapat melakukan tindakan dan mengakses sumber daya. Entitas ini biasanya merupakan pengguna root untuk Akun AWS, peran IAM, atau pengguna. Untuk informasi selengkapnya, lihat Prinsip dalam istilah dan konsep Peran dalam dokumentasi IAM.

Privasi oleh Desain

Pendekatan dalam rekayasa sistem yang memperhitungkan privasi di seluruh proses rekayasa.

zona host pribadi

Container yang menyimpan informasi tentang bagaimana Anda ingin Amazon Route 53 merespons kueri DNS untuk domain dan subdomainnya dalam satu atau beberapa VPC. Untuk

informasi selengkapnya, lihat Bekerja dengan zona yang dihosting pribadi di dokumentasi Route 53.

kontrol proaktif

Kontrol keamanan yang dirancang untuk mencegah penyebaran sumber daya yang tidak sesuai. Kontrol ini memindai sumber daya sebelum disediakan. Jika sumber daya tidak sesuai dengan kontrol, maka itu tidak disediakan. Untuk informasi selengkapnya, lihat panduan referensi Kontrol dalam AWS Control Tower dokumentasi dan lihat Kontrol proaktif dalam Menerapkan kontrol keamanan pada AWS.

manajemen siklus hidup produk (PLM)

Manajemen data dan proses untuk suatu produk di seluruh siklus hidupnya, mulai dari desain, pengembangan, dan peluncuran, melalui pertumbuhan dan kematangan, hingga penurunan dan penghapusan.

lingkungan produksi

Lihat lingkungan.

pengontrol logika yang dapat diprogram (PLC)

Di bidang manufaktur, komputer yang sangat andal dan mudah beradaptasi yang memantau mesin dan mengotomatiskan proses manufaktur.

pseudonimisasi

Proses penggantian pengidentifikasi pribadi dalam kumpulan data dengan nilai placeholder. Pseudonimisasi dapat membantu melindungi privasi pribadi. Data pseudonim masih dianggap sebagai data pribadi.

terbitkan/berlangganan (pub/sub)

Pola yang memungkinkan komunikasi asinkron antara layanan mikro untuk meningkatkan skalabilitas dan daya tanggap. Misalnya, dalam MES berbasis layanan mikro, layanan mikro dapat mempublikasikan pesan peristiwa ke saluran yang dapat berlangganan layanan mikro lainnya. Sistem dapat menambahkan layanan mikro baru tanpa mengubah layanan penerbitan.

Q

rencana kueri

Serangkaian langkah, seperti instruksi, yang digunakan untuk mengakses data dalam sistem database relasional SQL.

regresi rencana kueri

Ketika pengoptimal layanan database memilih rencana yang kurang optimal daripada sebelum perubahan yang diberikan ke lingkungan database. Hal ini dapat disebabkan oleh perubahan statistik, kendala, pengaturan lingkungan, pengikatan parameter kueri, dan pembaruan ke mesin database.

R

Matriks RACI

Lihat bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI).

ransomware

Perangkat lunak berbahaya yang dirancang untuk memblokir akses ke sistem komputer atau data sampai pembayaran dilakukan.

Matriks RASCI

Lihat bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI).

RCAC

Lihat kontrol akses baris dan kolom.

replika baca

Salinan database yang digunakan untuk tujuan read-only. Anda dapat merutekan kueri ke replika baca untuk mengurangi beban pada database utama Anda.

arsitek ulang

Lihat 7 Rs.

tujuan titik pemulihan (RPO)

Jumlah waktu maksimum yang dapat diterima sejak titik pemulihan data terakhir. Ini menentukan apa yang dianggap sebagai hilangnya data yang dapat diterima antara titik pemulihan terakhir dan gangguan layanan.

tujuan waktu pemulihan (RTO)

Penundaan maksimum yang dapat diterima antara gangguan layanan dan pemulihan layanan.

refactor

Lihat 7 Rs.

Wilayah

Kumpulan AWS sumber daya di wilayah geografis. Masing-masing Wilayah AWS terisolasi dan independen dari yang lain untuk memberikan toleransi kesalahan, stabilitas, dan ketahanan. Untuk informasi selengkapnya, lihat Menentukan Wilayah AWS akun yang dapat digunakan.

regresi

Teknik ML yang memprediksi nilai numerik. Misalnya, untuk memecahkan masalah “Berapa harga rumah ini akan dijual?” Model ML dapat menggunakan model regresi linier untuk memprediksi harga jual rumah berdasarkan fakta yang diketahui tentang rumah (misalnya, luas persegi).

rehost

Lihat 7 Rs.

melepaskan

Dalam proses penyebaran, tindakan mempromosikan perubahan pada lingkungan produksi.

memindahkan

Lihat 7 Rs.

memplatform ulang

Lihat 7 Rs.

pembelian kembali

Lihat 7 Rs.

ketahanan

Kemampuan aplikasi untuk melawan atau pulih dari gangguan. Ketersediaan tinggi dan pemulihan bencana adalah pertimbangan umum ketika merencanakan ketahanan di. AWS Cloud Untuk informasi lebih lanjut, lihat AWS Cloud Ketahanan.

kebijakan berbasis sumber daya

Kebijakan yang dilampirkan ke sumber daya, seperti bucket Amazon S3, titik akhir, atau kunci enkripsi. Jenis kebijakan ini menentukan prinsip mana yang diizinkan mengakses, tindakan yang didukung, dan kondisi lain yang harus dipenuhi.

matriks yang bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI)

Matriks yang mendefinisikan peran dan tanggung jawab untuk semua pihak yang terlibat dalam kegiatan migrasi dan operasi cloud. Nama matriks berasal dari jenis tanggung jawab yang didefinisikan dalam matriks: bertanggung jawab (R), akuntabel (A), dikonsultasikan (C), dan diinformasikan (I). Jenis dukungan (S) adalah opsional. Jika Anda menyertakan dukungan, matriks disebut matriks RASCI, dan jika Anda mengecualikannya, itu disebut matriks RACI.

kontrol responsif

Kontrol keamanan yang dirancang untuk mendorong remediasi efek samping atau penyimpangan dari garis dasar keamanan Anda. Untuk informasi selengkapnya, lihat Kontrol responsif dalam Menerapkan kontrol keamanan pada AWS.

melestarikan

Lihat 7 Rs.

pensiun

Lihat 7 Rs.

rotasi

Proses memperbarui rahasia secara berkala untuk membuatnya lebih sulit bagi penyerang untuk mengakses kredensil.

kontrol akses baris dan kolom (RCAC)

Penggunaan ekspresi SQL dasar dan fleksibel yang telah menetapkan aturan akses. RCAC terdiri dari izin baris dan topeng kolom.

RPO

Lihat tujuan titik pemulihan.

RTO

Lihat tujuan waktu pemulihan.

buku runbook

Satu set prosedur manual atau otomatis yang diperlukan untuk melakukan tugas tertentu. Ini biasanya dibangun untuk merampingkan operasi berulang atau prosedur dengan tingkat kesalahan yang tinggi.

D

SAML 2.0

Standar terbuka yang digunakan oleh banyak penyedia identitas (IdPs). Fitur ini memungkinkan sistem masuk tunggal gabungan (SSO), sehingga pengguna dapat masuk ke AWS Management Console atau memanggil operasi AWS API tanpa Anda harus membuat pengguna di IAM untuk semua orang di organisasi Anda. Untuk informasi lebih lanjut tentang federasi berbasis SAMP 2.0, lihat Tentang federasi berbasis SAMP 2.0 dalam dokumentasi IAM.

PENIPUAN

Lihat kontrol pengawasan dan akuisisi data.

SCP

Lihat kebijakan kontrol layanan.

Rahasia

Dalam AWS Secrets Manager, informasi rahasia atau terbatas, seperti kata sandi atau kredensil pengguna, yang Anda simpan dalam bentuk terenkripsi. Ini terdiri dari nilai rahasia dan metadata-nya. Nilai rahasia dapat berupa biner, string tunggal, atau beberapa string. Untuk informasi selengkapnya, lihat Apa yang ada di rahasia Secrets Manager? dalam dokumentasi Secrets Manager.

kontrol keamanan

Pagar pembatas teknis atau administratif yang mencegah, mendeteksi, atau mengurangi kemampuan pelaku ancaman untuk mengeksploitasi kerentanan keamanan. Ada empat jenis kontrol keamanan utama: preventif, detektif, responsif, dan proaktif.

pengerasan keamanan

Proses mengurangi permukaan serangan untuk membuatnya lebih tahan terhadap serangan. Ini dapat mencakup tindakan seperti menghapus sumber daya yang tidak lagi diperlukan, menerapkan praktik keamanan terbaik untuk memberikan hak istimewa paling sedikit, atau menonaktifkan fitur yang tidak perlu dalam file konfigurasi.

sistem informasi keamanan dan manajemen acara (SIEM)

Alat dan layanan yang menggabungkan sistem manajemen informasi keamanan (SIM) dan manajemen acara keamanan (SEM). Sistem SIEM mengumpulkan, memantau, dan menganalisis data dari server, jaringan, perangkat, dan sumber lain untuk mendeteksi ancaman dan pelanggaran keamanan, dan untuk menghasilkan peringatan.

otomatisasi respons keamanan

Tindakan yang telah ditentukan dan diprogram yang dirancang untuk secara otomatis merespons atau memulihkan peristiwa keamanan. Otomatisasi ini berfungsi sebagai kontrol keamanan detektif atau responsif yang membantu Anda menerapkan praktik terbaik AWS keamanan. Contoh tindakan respons otomatis termasuk memodifikasi grup keamanan VPC, menambal instans Amazon EC2, atau memutar kredensil.

enkripsi sisi server

Enkripsi data di tujuannya, oleh Layanan AWS yang menerimanya.

kebijakan kontrol layanan (SCP)

Kebijakan yang menyediakan kontrol terpusat atas izin untuk semua akun di organisasi. AWS Organizations SCP menentukan pagar pembatas atau menetapkan batasan pada tindakan yang dapat didelegasikan oleh administrator kepada pengguna atau peran. Anda dapat menggunakan SCP sebagai daftar izin atau daftar penolakan, untuk menentukan layanan atau tindakan mana yang diizinkan atau dilarang. Untuk informasi selengkapnya, lihat Kebijakan kontrol layanan dalam AWS Organizations dokumentasi.

titik akhir layanan

URL titik masuk untuk file Layanan AWS. Anda dapat menggunakan endpoint untuk terhubung secara terprogram ke layanan target. Untuk informasi selengkapnya, lihat Layanan AWS titik akhir di Referensi Umum AWS.

perjanjian tingkat layanan (SLA)

Perjanjian yang menjelaskan apa yang dijanjikan tim TI untuk diberikan kepada pelanggan mereka, seperti waktu kerja dan kinerja layanan.

indikator tingkat layanan (SLI)

Pengukuran aspek kinerja layanan, seperti tingkat kesalahan, ketersediaan, atau throughputnya.

tujuan tingkat layanan (SLO)

Metrik target yang mewakili kesehatan layanan, yang diukur dengan indikator tingkat layanan.

model tanggung jawab bersama

Model yang menjelaskan tanggung jawab yang Anda bagikan AWS untuk keamanan dan kepatuhan cloud. AWS bertanggung jawab atas keamanan cloud, sedangkan Anda bertanggung jawab atas keamanan di cloud. Untuk informasi selengkapnya, lihat Model tanggung jawab bersama.

SIEM

Lihat informasi keamanan dan sistem manajemen acara.

titik kegagalan tunggal (SPOF)

Kegagalan dalam satu komponen penting dari aplikasi yang dapat mengganggu sistem.

SLA

Lihat perjanjian tingkat layanan.

SLI

Lihat indikator tingkat layanan.

SLO

Lihat tujuan tingkat layanan.

split-and-seed model

Pola untuk menskalakan dan mempercepat proyek modernisasi. Ketika fitur baru dan rilis produk didefinisikan, tim inti berpisah untuk membuat tim produk baru. Ini membantu meningkatkan kemampuan dan layanan organisasi Anda, meningkatkan produktivitas pengembang, dan

mendukung inovasi yang cepat. Untuk informasi lebih lanjut, lihat Pendekatan bertahap untuk memodernisasi aplikasi di. AWS Cloud

SPOF

Lihat satu titik kegagalan.

skema bintang

Struktur organisasi database yang menggunakan satu tabel fakta besar untuk menyimpan data transaksional atau terukur dan menggunakan satu atau lebih tabel dimensi yang lebih kecil untuk menyimpan atribut data. Struktur ini dirancang untuk digunakan dalam gudang data atau untuk tujuan intelijen bisnis.

pola ara pencekik

Pendekatan untuk memodernisasi sistem monolitik dengan menulis ulang secara bertahap dan mengganti fungsionalitas sistem sampai sistem warisan dapat dinonaktifkan. Pola ini menggunakan analogi pohon ara yang tumbuh menjadi pohon yang sudah mapan dan akhirnya mengatasi dan menggantikan inangnya. Pola ini diperkenalkan oleh Martin Fowler sebagai cara untuk mengelola risiko saat menulis ulang sistem monolitik. Untuk contoh cara menerapkan pola ini, lihat Memodernisasi layanan web Microsoft ASP.NET (ASMX) lama secara bertahap menggunakan container dan Amazon API Gateway.

subnet

Rentang alamat IP dalam VPC Anda. Subnet harus berada di Availability Zone tunggal.

kontrol pengawasan dan akuisisi data (SCADA)

Di bidang manufaktur, sistem yang menggunakan perangkat keras dan perangkat lunak untuk memantau aset fisik dan operasi produksi.

enkripsi simetris

Algoritma enkripsi yang menggunakan kunci yang sama untuk mengenkripsi dan mendekripsi data.

pengujian sintetis

Menguji sistem dengan cara yang mensimulasikan interaksi pengguna untuk mendeteksi potensi masalah atau untuk memantau kinerja. Anda dapat menggunakan Amazon CloudWatch Synthetics untuk membuat tes ini.

T

tag

Pasangan nilai kunci yang bertindak sebagai metadata untuk mengatur sumber daya Anda. AWS Tanda dapat membantu Anda mengelola, mengidentifikasi, mengatur, dan memfilter sumber daya. Untuk informasi selengkapnya, lihat Menandai AWS sumber daya Anda.

variabel target

Nilai yang Anda coba prediksi dalam ML yang diawasi. Ini juga disebut sebagai variabel hasil. Misalnya, dalam pengaturan manufaktur, variabel target bisa menjadi cacat produk.

daftar tugas

Alat yang digunakan untuk melacak kemajuan melalui runbook. Daftar tugas berisi ikhtisar runbook dan daftar tugas umum yang harus diselesaikan. Untuk setiap tugas umum, itu termasuk perkiraan jumlah waktu yang dibutuhkan, pemilik, dan kemajuan.

lingkungan uji

Lihat lingkungan.

pelatihan

Untuk menyediakan data bagi model ML Anda untuk dipelajari. Data pelatihan harus berisi jawaban yang benar. Algoritma pembelajaran menemukan pola dalam data pelatihan yang memetakan atribut data input ke target (jawaban yang ingin Anda prediksi). Ini menghasilkan model ML yang menangkap pola-pola ini. Anda kemudian dapat menggunakan model ML untuk membuat prediksi pada data baru yang Anda tidak tahu targetnya.

gerbang transit

Hub transit jaringan yang dapat Anda gunakan untuk menghubungkan VPC dan jaringan lokal Anda. Untuk informasi selengkapnya, lihat Apa itu gateway transit dalam AWS Transit Gateway dokumentasi.

alur kerja berbasis batang

Pendekatan di mana pengembang membangun dan menguji fitur secara lokal di cabang fitur dan kemudian menggabungkan perubahan tersebut ke cabang utama. Cabang utama kemudian dibangun untuk pengembangan, praproduksi, dan lingkungan produksi, secara berurutan.

akses tepercaya

Memberikan izin ke layanan yang Anda tentukan untuk melakukan tugas di organisasi Anda di dalam AWS Organizations dan di akunnya atas nama Anda. Layanan tepercaya menciptakan peran terkait layanan di setiap akun, ketika peran itu diperlukan, untuk melakukan tugas manajemen untuk Anda. Untuk informasi selengkapnya, lihat Menggunakan AWS Organizations dengan AWS layanan lain dalam AWS Organizations dokumentasi.

penyetelan

Untuk mengubah aspek proses pelatihan Anda untuk meningkatkan akurasi model ML. Misalnya, Anda dapat melatih model ML dengan membuat set pelabelan, menambahkan label, dan kemudian mengulangi langkah-langkah ini beberapa kali di bawah pengaturan yang berbeda untuk mengoptimalkan model.

tim dua pizza

Sebuah DevOps tim kecil yang bisa Anda beri makan dengan dua pizza. Ukuran tim dua pizza memastikan peluang terbaik untuk berkolaborasi dalam pengembangan perangkat lunak.

U

waswas

Sebuah konsep yang mengacu pada informasi yang tidak tepat, tidak lengkap, atau tidak diketahui yang dapat merusak keandalan model ML prediktif. Ada dua jenis ketidakpastian: ketidakpastian epistemik disebabkan oleh data yang terbatas dan tidak lengkap, sedangkan ketidakpastian aleatorik disebabkan oleh kebisingan dan keacakan yang melekat dalam data. Untuk informasi lebih lanjut, lihat panduan Mengukur ketidakpastian dalam sistem pembelajaran mendalam.

tugas yang tidak terdiferensiasi

Juga dikenal sebagai angkat berat, pekerjaan yang diperlukan untuk membuat dan mengoperasikan aplikasi tetapi itu tidak memberikan nilai langsung kepada pengguna akhir atau memberikan keunggulan kompetitif. Contoh tugas yang tidak terdiferensiasi termasuk pengadaan, pemeliharaan, dan perencanaan kapasitas.

lingkungan atas

Lihat lingkungan.

V

menyedot debu

Operasi pemeliharaan database yang melibatkan pembersihan setelah pembaruan tambahan untuk merebut kembali penyimpanan dan meningkatkan kinerja.

kendali versi

Proses dan alat yang melacak perubahan, seperti perubahan kode sumber dalam repositori.

Peering VPC

Koneksi antara dua VPC yang memungkinkan Anda merutekan lalu lintas dengan menggunakan alamat IP pribadi. Untuk informasi selengkapnya, lihat Apa itu peering VPC di dokumentasi VPC Amazon.

kerentanan

Kelemahan perangkat lunak atau perangkat keras yang membahayakan keamanan sistem.

W

cache hangat

Cache buffer yang berisi data terkini dan relevan yang sering diakses. Instance database dapat membaca dari cache buffer, yang lebih cepat daripada membaca dari memori utama atau disk.

data hangat

Data yang jarang diakses. Saat menanyakan jenis data ini, kueri yang cukup lambat biasanya dapat diterima.

fungsi jendela

Fungsi SQL yang melakukan perhitungan pada sekelompok baris yang berhubungan dengan catatan saat ini. Fungsi jendela berguna untuk memproses tugas, seperti menghitung rata-rata bergerak atau mengakses nilai baris berdasarkan posisi relatif dari baris saat ini.

beban kerja

Kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi yang dihadapi pelanggan atau proses backend.

aliran kerja

Grup fungsional dalam proyek migrasi yang bertanggung jawab atas serangkaian tugas tertentu. Setiap alur kerja independen tetapi mendukung alur kerja lain dalam proyek. Misalnya, alur kerja portofolio bertanggung jawab untuk memprioritaskan aplikasi, perencanaan gelombang, dan mengumpulkan metadata migrasi. Alur kerja portofolio mengirimkan aset ini ke alur kerja migrasi, yang kemudian memigrasikan server dan aplikasi.

CACING

Lihat menulis sekali, baca banyak.

WQF

Lihat Kerangka Kualifikasi Beban Kerja AWS.

tulis sekali, baca banyak (WORM)

Model penyimpanan yang menulis data satu kali dan mencegah data dihapus atau dimodifikasi. Pengguna yang berwenang dapat membaca data sebanyak yang diperlukan, tetapi mereka tidak dapat mengubahnya. Infrastruktur penyimpanan data ini dianggap tidak dapat diubah.

Z

eksploitasi zero-day

Serangan, biasanya malware, yang memanfaatkan kerentanan zero-day.

kerentanan zero-day

Cacat atau kerentanan yang tak tanggung-tanggung dalam sistem produksi. Aktor ancaman dapat menggunakan jenis kerentanan ini untuk menyerang sistem. Pengembang sering menyadari kerentanan sebagai akibat dari serangan tersebut.

aplikasi zombie

Aplikasi yang memiliki CPU rata-rata dan penggunaan memori di bawah 5 persen. Dalam proyek migrasi, adalah umum untuk menghentikan aplikasi ini.

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.