Pemodelan data dengan Amazon DynamoDB

# AWS Bimbingan Preskriptif

# AWS Bimbingan Preskriptif: Pemodelan data dengan Amazon DynamoDB

# Table of Contents

# Pemodelan data dengan Amazon DynamoDB

## Proses, templat, dan praktik terbaik

Amazon Web Services (kontributor)

Desember 2023 (riwayat dokumen)

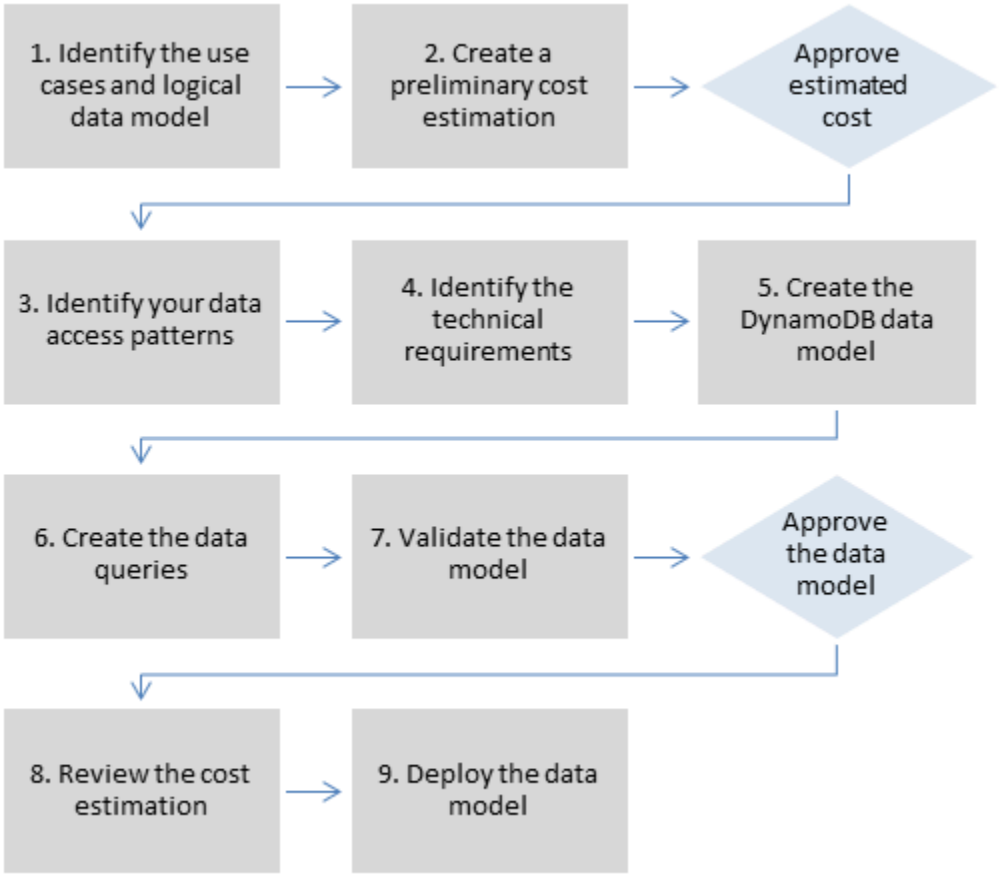
Database NoSQL menyediakan skema fleksibel untuk membangun aplikasi modern. Mereka diakui secara luas karena kemudahan pengembangan, fungsionalitas, dan kinerja dalam skala besar. Amazon DynamoDB menyediakan kinerja yang cepat dan dapat diprediksi dengan skalabilitas yang mulus untuk database NoSQL di Amazon Web Services () Cloud.AWS Sebagai layanan database yang dikelola sepenuhnya, DynamoDB membantu Anda menurunkan beban administratif pengoperasian dan penskalaan database terdistribusi. Anda tidak perlu khawatir tentang penyediaan perangkat keras, penyiapan dan konfigurasi, replikasi, penambalan perangkat lunak, atau penskalaan cluster.

Desain skema NoSQL memerlukan pendekatan yang berbeda dari desain sistem manajemen basis data relasional tradisional (RDBMS). Model data RDBMS berfokus pada struktur data dan hubungannya dengan data lain. Pemodelan data NoSQL berfokus pada pola akses, atau bagaimana aplikasi akan mengkonsumsi data, sehingga menyimpan data dengan cara yang mendukung operasi kueri langsung. Untuk RDBMS seperti Microsoft SQL Server atau IBM Db2, Anda dapat membuat model data yang dinormalisasi tanpa memikirkan banyak pola akses. Anda dapat memperluas model data untuk mendukung pola dan kueri Anda nanti.

Panduan ini menyajikan proses pemodelan data untuk menggunakan DynamoDB yang menyediakan persyaratan fungsional, kinerja, dan biaya efektif. Panduan ini adalah untuk insinyur database yang berencana untuk menggunakan DynamoDB sebagai database operasional untuk aplikasi mereka yang berjalan pada. AWS AWS Layanan Profesional telah menggunakan proses yang direkomendasikan untuk membantu perusahaan perusahaan dengan pemodelan data DynamoDB untuk berbagai kasus penggunaan dan beban kerja.

# Alur proses pemodelan data

Kami merekomendasikan proses berikut saat memodelkan data menggunakan Amazon DynamoDB. Langkah-langkahnya dibahas secara rinci nanti dalam panduan ini.




# Matriks RACI

Beberapa organisasi menggunakan matriks penugasan tanggung jawab (juga dikenal sebagai matriks RACI) untuk menggambarkan berbagai peran yang terlibat dalam satu proyek atau proses bisnis tertentu. Panduan ini menyajikan matriks RACI yang disarankan yang dapat membantu organisasi Anda mengidentifikasi orang yang tepat dan tanggung jawab yang tepat untuk proses pemodelan data DynamoDB. Untuk setiap langkah dalam proses, daftar pemangku kepentingan dan keterlibatan mereka:

- R — bertanggung jawab untuk menyelesaikan langkah

- A — bertanggung jawab untuk menyetujui dan menandatangani pekerjaan
- C — dikonsultasikan untuk memberikan masukan untuk tugas
- Saya — diberitahu tentang kemajuan, tetapi tidak terlibat langsung dalam tugas

Bergantung pada struktur organisasi dan tim proyek Anda, peran dalam matriks RACI berikut dapat dilakukan oleh pemangku kepentingan yang sama. Dalam beberapa situasi, pemangku kepentingan bertanggung jawab dan bertanggung jawab atas langkah-langkah tertentu. Misalnya, insinyur database dapat bertanggung jawab untuk membuat dan menyetujui model data, karena ini adalah area domain mereka.

| Langkah proses | Pengguna bisnis | Analis bisnis | Arsitek solusi | Insinyur basis data | Pengembang aplikasi | DevOps insinyur |
|---|---|---|---|---|---|---|
| 1. Identifikasi kasus penggunaan dan model data logis | C | R/A | I | R | | |
| 2. Buat estimasi biaya awal | C | A | I | R | | |
| 3. Identifikasi pola akses data Anda | C | A | I | R | | |
| 4. Identifikasi persyaratan teknis | C | C | A | R | | |
| 5. Buat model data DynamoDB | I | I | I | R/A | | |

| Langkah proses | Pengguna bisnis | Analis bisnis | Arsitek solusi | Insinyur basis data | Pengembang aplikasi | DevOps insinyur |
|---|---|---|---|---|---|---|
| 6. Buat kueri data | I | I | I | R/A | R | |
| 7. Validasi model data | A | R | I | C | | |
| 8. Tinjau estimasi biaya | C | A | I | R | | |
| 9. Menyebarkan model data DynamoDB | I | I | C | C | | R/A |

# Langkah-langkah proses pemodelan data

Bagian ini merinci setiap langkah proses pemodelan data yang direkomendasikan untuk Amazon DynamoDB.

Topik



## Langkah 1. Identifikasi kasus penggunaan dan model data logis

### Tujuan

- Kumpulkan kebutuhan bisnis dan gunakan kasus yang memerlukan database NoSQL.
- Tentukan model data logis dengan menggunakan diagram entity-relationship (ER).

### Proses

- Analis bisnis mewawancarai pengguna bisnis untuk mengidentifikasi kasus penggunaan dan hasil yang diharapkan.
- Database engineer menciptakan model data konseptual.
- Insinyur database menciptakan model data logis.
- Insinyur database mengumpulkan informasi tentang ukuran item, volume data, dan throughput baca dan tulis yang diharapkan.

## Alat dan sumber daya

- Penilaian persyaratan bisnis (lihat templat)
- Matriks pola akses (lihat templat)
- Alat pilihan Anda untuk membuat diagram

## RACI

| Pengguna bisnis | Analis bisnis | Arsitek solusi | Insinyur basis data | Pengembang aplikasi | DevOps insinyur |
|---|---|---|---|---|---|
| C | R/A | I | R | | |

## Output

- Kasus penggunaan dan persyaratan bisnis yang terdokumentasi
- Model data logis (diagram ER)

# Langkah 2. Buat estimasi biaya awal

## Tujuan

- Kembangkan estimasi biaya awal untuk DynamoDB.

## Proses

- Insinyur database membuat analisis biaya awal menggunakan informasi yang tersedia dan contoh yang disajikan pada halaman harga DynamoDB.
  - Buat perkiraan biaya untuk kapasitas sesuai permintaan (lihat contoh).
  - Buat estimasi biaya untuk kapasitas yang disediakan (lihat contoh).
    - Untuk model kapasitas yang disediakan, dapatkan estimasi biaya dari kalkulator dan terapkan diskon untuk kapasitas cadangan.
  - Bandingkan perkiraan biaya dari dua model kapasitas.

  - Buat estimasi untuk semua lingkungan (Dev, Prod, QA).
- Analis bisnis meninjau dan menyetujui atau menolak perkiraan biaya awal.

### Alat dan sumber daya

- AWS Kalkulator Harga

### RACI

| Pengguna bisnis | Analis bisnis | Arsitek solusi | Insinyur basis data | Pengembang aplikasi | DevOps insinyur |
|---|---|---|---|---|---|
| C | A | I | R | | |

### Output

- Estimasi biaya awal

## Langkah 3. Identifikasi pola akses data Anda

Pola akses atau pola kueri menentukan bagaimana pengguna dan sistem mengakses data untuk memenuhi kebutuhan bisnis.

### Tujuan

- Dokumentasikan pola akses data.

### Proses

- Insinyur basis data dan analis bisnis mewawancarai pengguna akhir untuk mengidentifikasi bagaimana data akan ditanyakan menggunakan templat matriks pola akses data.
  - Untuk aplikasi baru, mereka meninjau cerita pengguna tentang aktivitas dan tujuan. Mereka mendokumentasikan kasus penggunaan dan menganalisis pola akses yang diperlukan oleh kasus penggunaan.

  - Untuk aplikasi yang ada, mereka menganalisis log kueri untuk mengetahui bagaimana orang saat ini menggunakan sistem dan untuk mengidentifikasi pola akses utama.
- Database engineer mengidentifikasi properti berikut dari pola akses:
  - Ukuran data: Mengetahui berapa banyak data yang akan disimpan dan diminta sekaligus membantu menentukan cara paling efektif untuk mempartisi data (lihat posting blog).
  - Bentuk data: Alih-alih membentuk kembali data saat kueri diproses (seperti yang dilakukan sistem RDBMS), basis data NoSQL mengatur data sehingga bentuknya dalam basis data tersebut sesuai dengan apa yang akan dikueri. Ini adalah faktor kunci dalam meningkatkan kecepatan dan skalabilitas.
  - Kecepatan data: DynamoDB menskalakan dengan meningkatkan jumlah partisi fisik yang tersedia untuk memproses kueri, dan dengan mendistribusikan data secara efisien ke seluruh partisi tersebut. Mengetahui beban kueri puncak sebelumnya dapat membantu menentukan cara mempartisi data ke I/O kapasitas penggunaan terbaik.
- Pengguna bisnis memprioritaskan pola akses atau kueri.
  - Kueri prioritas biasanya adalah kueri yang paling banyak digunakan atau paling relevan. Penting juga untuk mengidentifikasi kueri yang membutuhkan latensi respons yang lebih rendah.

## Alat dan sumber daya

- Matriks pola akses (lihat templat)
- Memilih Kunci AWS Partisi DynamoDB yang Tepat (Blog Database)
- Desain NoSQL untuk DynamoDB (dokumentasi DynamoDB)

## RACI

| Pengguna bisnis | Analis bisnis | Arsitek solusi | Insinyur basis data | Pengembang aplikasi | DevOps insinyur |
|---|---|---|---|---|---|
| C | A | I | R | | |

## Output

- Matriks pola akses data

## Contoh

| Pola akses | Prioritas | Membaca atau menulis | Deskripsi | Jenis (item tunggal, beberapa item, atau semua) | Atribut kunci | Filter | Pemesanan hasil |
|---|---|---|---|---|---|---|---|
| Buat profil pengguna | Tinggi | Tulis | Pengguna membuat profil baru | Item tunggal | nama pengguna | N/A | N/A |
| Perbarui profil pengguna | Sedang | Tulis | Pengguna memperbarui profil mereka | Item tunggal | nama pengguna | Nama pengguna = pengguna saat ini | N/A |

# Langkah 4. Identifikasi persyaratan teknis

## Tujuan

- Kumpulkan persyaratan teknis untuk database DynamoDB.

## Proses

- Analis bisnis mewawancarai pengguna bisnis dan DevOps tim untuk mengumpulkan persyaratan teknis dengan menggunakan kuesioner penilaian.

## Alat dan sumber daya

- Penilaian persyaratan teknis (lihat contoh kuesioner)

### RACI

| Pengguna bisnis | Analis bisnis | Arsitek solusi | Insinyur basis data | Pengembang aplikasi | DevOps insinyur |
|---|---|---|---|---|---|
| C | C | A | R | | |

### Output

- Dokumen persyaratan teknis

## Langkah 5. Buat model data DynamoDB

### Tujuan

- Buat model data DynamoDB.

### Proses

- Insinyur database mengidentifikasi berapa banyak tabel yang diperlukan untuk setiap kasus penggunaan. Sebaiknya pertahankan tabel sesedikit mungkin dalam aplikasi DynamoDB.
- Berdasarkan pola akses yang paling umum, identifikasi kunci utama yang dapat menjadi salah satu dari dua jenis: kunci primer dengan kunci partisi yang mengidentifikasi data, atau kunci primer dengan kunci partisi dan kunci sortir. Kunci sortir adalah kunci sekunder untuk pengelompokan dan pengorganisasian data sehingga dapat ditanyakan dalam partisi secara efisien. Anda dapat menggunakan kunci sortir untuk menentukan hubungan hierarkis dalam data Anda yang dapat Anda kueri di setiap tingkat hierarki (lihat posting blog).
  - Desain kunci partisi
    - Tentukan kunci partisi dan evaluasi distribusinya.
    - Identifikasi kebutuhan penulisan sharding untuk mendistribusikan beban kerja secara merata.
  - Desain kunci urutan
    - Identifikasi kunci sortir.
    - Identifikasi kebutuhan untuk kunci sortir komposit.

    - Identifikasi kebutuhan untuk kontrol versi.
- Berdasarkan pola akses, identifikasi indeks sekunder untuk memenuhi persyaratan kueri.
    - Identifikasi kebutuhan untuk indeks sekunder lokal (LSIs). Ini adalah indeks yang memiliki kunci partisi yang sama dengan tabel dasar, tetapi kunci pengurutan yang berbeda.
        - Untuk tabel dengan LSIs, ada batas ukuran 10 GB per nilai kunci partisi. Sebuah tabel dengan LSIs dapat menyimpan sejumlah item, selama ukuran total untuk salah satu nilai kunci partisi tidak melebihi 10 GB.
    - Identifikasi kebutuhan untuk indeks sekunder global (GSIs). Ini adalah indeks yang memiliki kunci partisi dan kunci pengurutan yang dapat berbeda dari yang ada di tabel dasar (lihat posting blog).
    - Tentukan proyeksi indeks. Pertimbangkan untuk memproyeksikan lebih sedikit atribut untuk meminimalkan ukuran item yang ditulis ke indeks. Pada langkah ini, Anda harus menentukan apakah Anda ingin menggunakan yang berikut ini:
        - Indeks jarang
        - Kueri agregasi terwujud
        - GSI kelebihan beban
        - Sharding GSI
        - Replika yang akhirnya konsisten menggunakan GSI
- Insinyur database menentukan apakah data akan mencakup item besar. Jika demikian, mereka merancang solusi dengan menggunakan kompresi atau dengan menyimpan data di Amazon Simple Storage Service (Amazon S3).
- Insinyur database menentukan apakah data deret waktu akan dibutuhkan. Jika demikian, mereka menggunakan pola desain deret waktu untuk memodelkan data.
- Insinyur basis data menentukan apakah model ER mencakup many-to-many hubungan. Jika demikian, mereka menggunakan pola desain daftar kedekatan untuk memodelkan data.

## Alat dan sumber daya

- NoSQL Workbench untuk Amazon DynamoDB - Menyediakan pemodelan data, visualisasi data, dan pengembangan kueri dan fitur pengujian untuk membantu Anda mendesain database DynamoDB
- Desain NoSQL untuk DynamoDB (dokumentasi DynamoDB)
- Memilih Kunci AWS Partisi DynamoDB yang Tepat (Blog Database)

- Praktik terbaik untuk menggunakan indeks sekunder di DynamoDB (dokumentasi DynamoDB)
- Bagaimana merancang indeks AWS sekunder global Amazon DynamoDB (blog Database)

## RACI

| Pengguna bisnis | Analis bisnis | Arsitek solusi | Insinyur basis data | Pengembang aplikasi | DevOps insinyur |
|---|---|---|---|---|---|
| I | I | I | R/A | | |

## Output

- Skema tabel DynamoDB yang memenuhi pola dan persyaratan akses Anda

## Contoh

Tangkapan layar berikut menunjukkan NoSQL Workbench.

RetailDatabase | GSI: BundleProducts-VendorProducts-CustomerOrdersByOrderId | GSI: VendorOrdersByStatusDate-ProductInventory | GSI: ProductCatalog-CustomerOrdersByProduct

| Primary Key | | Attributes | | | | | |
|---|---|---|---|---|---|---|---|
| Partition Key: pk | Sort Key: sk | | | | | | |
| P1 | B1 | GSI1-PK | GSI1-SK | name | desc | | |
| | | B1 | P1 | The Tiki Bundle | Everything you need for an island theme party. | | |
| P4 | B2 | GSI1-PK | GSI1-SK | name | desc | | |
| | | B2 | P4 | Tiki Bar Set | Be the Mai Tai master with your very own Tiki Bar. | | |
| P2 | B1 | name | desc | qty | GSI1-PK | GSI1-SK | location |
| | | Tiki Torch | Bamboo tiki torch, 4 ft | 6 | B1 | P2 | W1-A9-S10-B52 |
| | B2 | name | desc | qty | GSI1-PK | GSI1-SK | location |
| | | Tiki Torch | Bamboo tiki torch, 4 ft | 2 | B2 | P2 | W1-A9-S10-B52 |
| | P2 | name | desc | qty | location | reorderAt | GSI3-SK |
| | | Tiki Torch | Bamboo tiki torch, 4 ft | 656 | W1-A9-S10-B52 | 100 | /GardenOutdoor/OutdoorDecor/Lighting/LanternsT |
| | B1 | name | desc | qty | GSI1-PK | GSI1-SK | location |
| | | Tiki Statue - Pele | Tiki of the Hawaiian Fire Goddess Pele, 5 ft. | 1 | B1 | P3 | W1-A15-S6-B27 |

# Langkah 6. Buat kueri data

## Tujuan

- Buat kueri utama untuk memvalidasi model data.

## Proses

- Insinyur database secara manual membuat tabel DynamoDB di Wilayah atau di AWS komputer mereka (DynamoDB Lokal).
- Insinyur database menambahkan data sampel ke tabel DynamoDB.
- Insinyur database membangun aspek menggunakan NoSQL Workbench untuk Amazon DynamoDB atau SDK for AWS Java atau Python untuk membuat contoh kueri (lihat posting blog).

  Aspek seperti tampilan tabel DynamoDB.
- Insinyur basis data dan pengembang cloud membuat kueri sampel dengan menggunakan AWS Command Line Interface (AWS CLI) atau AWS SDK untuk bahasa pilihan.

## Alat dan sumber daya

- AWS Akun aktif, untuk mendapatkan akses ke konsol DynamoDB
- DynamoDB Local (opsional), jika Anda ingin membangun database di komputer Anda tanpa mengakses layanan web DynamoDB
- NoSQL Workbench untuk Amazon DynamoDB (unduhan dan dokumentasi)
- AWS SDK dalam bahasa pilihan Anda (JavaScript, Python, PHP, .NET, Ruby, Java, Go, Node.js, C ++, dan SAP ABAP)

## RACI

| Pengguna bisnis | Analis bisnis | Arsitek solusi | Insinyur basis data | Pengembang aplikasi | DevOps insinyur |
|---|---|---|---|---|---|
| I | I | I | R/A | R | |

### Output

- Kode untuk menanyakan tabel DynamoDB

### Contoh

- Contoh DynamoDB menggunakan AWS SDK for Java
- Contoh Python
- JavaScriptcontoh

## Langkah 7. Validasi model data

### Tujuan

- Pastikan bahwa model data akan memenuhi kebutuhan Anda.

### Proses

- Insinyur database mengisi tabel DynamoDB dengan data sampel.
- Database engineer menjalankan kode untuk query tabel DynamoDB.
- Insinyur database mengumpulkan hasil kueri.
- Insinyur database mengumpulkan metrik kinerja kueri.
- Pengguna bisnis memvalidasi bahwa hasil kueri memenuhi kebutuhan bisnis.
- Analis bisnis memvalidasi persyaratan teknis.

### Alat dan sumber daya

- AWS Akun aktif, untuk mendapatkan akses ke konsol DynamoDB
- DynamoDB Local (opsional), jika Anda ingin membangun database di komputer Anda tanpa mengakses layanan web DynamoDB
- AWS SDK dalam bahasa pilihan Anda

### RACI

| Pengguna bisnis | Analis bisnis | Arsitek solusi | Insinyur basis data | Pengembang aplikasi | DevOps insinyur |
|---|---|---|---|---|---|
| A | R | I | C | | |

### Output

- Model data yang disetujui

## Langkah 8. Tinjau estimasi biaya

### Tujuan

- Tentukan model kapasitas dan perkirakan biaya DynamoDB untuk menyempurnakan estimasi biaya dari langkah 2.
- Dapatkan persetujuan keuangan akhir dari analis bisnis dan pemangku kepentingan.

### Proses

- Insinyur database mengidentifikasi estimasi volume data.
- Insinyur database mengidentifikasi persyaratan transfer data.
- Insinyur basis data mendefinisikan unit kapasitas baca dan tulis yang diperlukan.
- Analis bisnis memutuskan antara model kapasitas sesuai permintaan dan yang disediakan.
- Insinyur basis data mengidentifikasi kebutuhan untuk penskalaan otomatis DynamoDB.
- Insinyur database memasukkan parameter dalam alat Kalkulator Bulanan Sederhana.
- Insinyur basis data menyajikan estimasi harga akhir kepada pemangku kepentingan bisnis.
- Analis bisnis dan pemangku kepentingan menyetujui atau menolak solusi.

### Alat dan sumber daya

- AWS Kalkulator Harga

### RACI

| Pengguna bisnis | Analis bisnis | Arsitek solusi | Insinyur basis data | Pengembang aplikasi | DevOps insinyur |
|---|---|---|---|---|---|
| C | A | I | R | | |

### Output

- Model kapasitas
- Estimasi biaya yang direvisi

## Langkah 9. Menyebarkan model data

### Tujuan

- Menyebarkan tabel DynamoDB (atau tabel) ke. AWS Region

### Proses

- DevOps arsitek membuat CloudFormation template atau infrastruktur lain sebagai kode (IAc) alat untuk tabel DynamoDB (atau tabel). CloudFormation menyediakan cara otomatis untuk menyediakan dan mengkonfigurasi tabel Anda dan sumber daya terkait.

### Alat dan sumber daya

- CloudFormation

### RACI

| Pengguna bisnis | Analis bisnis | Arsitek solusi | Insinyur basis data | Pengembang aplikasi | DevOps insinyur |
|---|---|---|---|---|---|
| I | I | C | C | | R/A |

## Output

- AWS CloudFormation Template

## Contoh

```
mySecondDDBTable:
   Type: AWS::DynamoDB::
   Table DependsOn: "myFirstDDBTable"
   Properties:
   AttributeDefinitions:
      -  AttributeName: "ArtistId"
         AttributeType: "S"
      -  AttributeName: "Concert"
         AttributeType: "S"
      -  AttributeName: "TicketSales"
         AttributeType: "S"
   KeySchema:
      -  AttributeName: "ArtistId"
         KeyType: "HASH"
      -  AttributeName: "Concert"
         KeyType: "RANGE"
   ProvisionedThroughput:
      ReadCapacityUnits:
         Ref: "ReadCapacityUnits"
      WriteCapacityUnits:
         Ref: "WriteCapacityUnits"
   GlobalSecondaryIndexes:
      -  IndexName: "myGSI"
         KeySchema:
            - AttributeName: "TicketSales"
            KeyType: "HASH"
         Projection:
            ProjectionType: "KEYS_ONLY"
         ProvisionedThroughput:
         ReadCapacityUnits:
            Ref: "ReadCapacityUnits"
         WriteCapacityUnits:
            Ref: "WriteCapacityUnits"
   Tags:
      -  Key: mykey
```

```
        Value: myvalue
```

# Template

Template yang disediakan di bagian ini didasarkan pada Data Pemodelan Game Player dengan Amazon DynamoDB di situs web. AWS

> **Note**
>
> Tabel di bagian ini menggunakan MM sebagai singkatan untuk juta, dan K sebagai singkatan dari ribu.

Topik



## Templat penilaian persyaratan bisnis

Berikan deskripsi untuk kasus penggunaan:

> **Deskripsi**
>
> Bayangkan Anda sedang membangun game multiplayer online. Dalam permainan Anda, kelompok yang terdiri dari 50 pemain bergabung dalam sesi untuk bermain game, yang biasanya membutuhkan waktu sekitar 30 menit untuk bermain. Selama permainan, Anda harus memperbarui catatan pemain tertentu untuk menunjukkan jumlah waktu pemain telah bermain, statistik mereka, atau apakah mereka memenangkan permainan. Pengguna ingin melihat game sebelumnya yang telah mereka mainkan, baik untuk melihat pemenang game atau menonton tayangan ulang dari setiap aksi game.

Berikan informasi tentang pengguna Anda:

| Pengguna | Deskripsi | Jumlah yang diharapkan |
| --- | --- | --- |
| Pemain permainan | Pemain game online. | 1 MM |

| | | |
|---|---|---|
| Tim pengembangan | Tim internal yang akan menggunakan statistik permainan untuk meningkatkan<br><br>pengalaman permainan. | 100 |

Berikan informasi tentang sumber data dan bagaimana data akan dicerna:

| Sumber | Deskripsi | Pengguna |
|---|---|---|
| Game online | Pemain game akan membuat profil dan memulai game baru. | Pemain permainan |
| Aplikasi game | Aplikasi game akan secara otomatis mengumpulkan statistik tentang game, seperti waktu mulai dan berakhir, jumlah pemain, posisi setiap pemain, dan peta untuk permainan. | |

Berikan informasi tentang bagaimana data akan dikonsumsi:

| Konsumen | Deskripsi | Pengguna |
|---|---|---|
| Game online | Pemain game akan melihat profil dan meninjau statistik permainan mereka. | Pemain permainan |
| Analisis data | Tim pengembangan game akan mengekstrak statistik game untuk analisis data dan untuk meningkatkan pengalaman pengguna. Data akan diekspor dari penyimpan | Tim pengembangan |

| | | |
|---|---|---|
| | an data dan diimpor ke Amazon S3 untuk mendukung analitik melalui aplikasi Spark. | |

Berikan daftar entitas dan bagaimana mereka diidentifikasi:

| Nama entitas | Deskripsi | Pengenal |
|---|---|---|
| Pemain Game | Menyimpan informasi seperti identifikasi, alamat, demografi , minat untuk setiap pengguna (gamer). | Nama Pengguna |
| Contoh Permainan | Memberikan informasi tentang setiap game yang dimainkan , termasuk pembuat, mulai, akhir, dan peta Yplayed. | ID permainan |
| Pemetaan Pengguna Game | Merupakan many-to-many hubungan antara pengguna dan game. | ID Game DAN Nama Pengguna |

Buat model ER untuk entitas:


Berikan statistik tingkat tinggi tentang entitas:

| Nama Entitas | Perkiraan # catatan | Ukuran rekam | Catatan |
|---|---|---|---|
| Pemain Game | 1 MM | < 1 KB | Platform game memiliki sekitar 1 MM pengguna. |
| Contoh Permainan | 6 MM<br>(100.000K/hari * 60 hari) | < 1 KB | Rata-rata, ada 100K game setiap hari. Kita perlu menyimpan 60 hari terakhir. |
| Pemetaan Pengguna Game | 300 MM<br>(Game 6 MM* 50 pemain) | < 1 KB | Rata-rata, setiap game memiliki 50 pemain yang perlu kita simpan informasi nya. |

# Templat penilaian persyaratan teknis

Berikan informasi tentang jenis konsumsi data:

| Jenis konsumsi data | Y/N | Deskripsi | Frekuensi |
| --- | --- | --- | --- |
| Akses aplikasi | Y | | |
| Gerbang API | Y | | |
| Streaming data | N | | |
| Proses Batch | N | | |
| ETL | N | | |
| Impor data | N | | |
| Deret waktu | N | | |

Memberikan informasi tentang jenis konsumsi data:

| Jenis konsumsi data | Y/N | Deskripsi | Frekuensi |
| --- | --- | --- | --- |
| Akses aplikasi | | | |
| Gerbang API | | | |
| Ekspor data | | | |
| Analisis data | | | |
| Agregasi data | | | |
| Pelaporan | | | |
| Cari | | | |
| Streaming data | | | |
| ETL | | | |

Berikan perkiraan volume data:

| Nama entitas | Perkiraan # catatan | Ukuran rekam | Volume data |
| --- | --- | --- | --- |
| Pemain Game | 1 MM | < 1 KB | ~ 1 GB<br>(1 MM * 1 KB) |
| Contoh Permainan | 6 MM<br>(100K/hari * 60 hari) | < 1 KB | ~ 6 GB<br>(6 MM * 1 KB) |
| Pemetaan Pengguna Game | 300 MM<br>(Game 6 MM* 50 pemain) | < 1 KB | ~ 300 GB<br>(300 MM * 1 KB) |

> **Note**
>
> Periode penyimpanan data adalah 60 hari. Setelah 60 hari, data harus disimpan di Amazon S3 untuk analitik, dengan menggunakan DynamoDB Time to Live (TTL) untuk secara otomatis memindahkan data dari DynamoDB ke Amazon S3.

Jawab pertanyaan-pertanyaan ini tentang pola waktu:

- Kerangka waktu apa aplikasi tersedia untuk pengguna (misalnya, 24/7 atau 9 pagi hingga 5 sore pada hari kerja)?
- Apakah ada puncak penggunaan di siang hari? Berapa jam? Berapa persentase penggunaan aplikasi?

Tentukan persyaratan throughput tulis:

| Nama entitas | Menulis/hari | Jam/hari | Menulis/detik |
| --- | --- | --- | --- |
| Pemain Game | 10.000 pembaruan | 18 | < 1 |
| Contoh Permainan | 300.000 | 18 | < 5 |

| | | | |
|---|---|---|---|
| Pemetaan Pengguna Game | 1.800.000.000 | 18 | ~ 27,777 |

> **Catatan**
>
> Operasi penulisan Game Player: 1 persen pengguna memperbarui profil mereka setiap hari, jadi kami mengharapkan 10.000 pembaruan untuk 1.000.000 pengguna.
> Operasi penulisan Instans Game: 100.000 game/hari. Untuk setiap game, kami memiliki setidaknya 3 operasi tulis — saat pembuatan, awal, dan akhir — jadi totalnya adalah 300.000 operasi tulis.
> Operasi penulisan Pemetaan Pengguna Game: 100.000 game/hari untuk setiap game dengan 50 pemain. Durasi permainan rata-rata adalah 30 menit, dan posisi gamer diperbarui setiap 5 detik. Kami memperkirakan rata-rata 360 pembaruan per gamer, jadi totalnya adalah 100.000 * 50 * 360 = 1.800.000.000 operasi tulis.

Tentukan persyaratan throughput baca:

| Nama entitas | Membaca/hari | Jam/hari | Membaca/detik |
|---|---|---|---|
| Pemain Game | 200.000 | 18 | ~ 3 |
| Contoh Permainan | 5.000.000 | 18 | ~ 77 |
| Pemetaan Pengguna Game | 1.800.000.000 | 18 | ~ 27,777 |

> **Catatan**
>
> Game Player membaca operasi: 20 persen pengguna memulai game, jadi 1 MM* 0,2 = 200.000.
> Operasi baca Instans Game: 100.000 game/hari. Untuk setiap game kami memiliki setidaknya 1 operasi baca per pemain, dan 50 pemain per game, jadi totalnya adalah 5.000.000 operasi baca.
> Operasi baca Pemetaan Pengguna Game: 100.000 game/hari untuk 50 pemain. Durasi permainan rata-rata adalah 30 menit, dan posisi gamer diperbarui setiap 5 detik. Kami

memperkirakan rata-rata 360 pembaruan per gamer, dan setiap pembaruan memerlukan operasi baca, jadi totalnya adalah 100.000 * 50 * 360 = 1.800.000.000 operasi baca.

Tentukan persyaratan latensi akses data:

| Operasi | 99 persentil | Latensi maksimum |
| --- | --- | --- |
| Baca | 30 ms | 100 ms |
| Menulis | 10 ms | 50 ms |

Tentukan persyaratan ketersediaan data:

| Persyaratan | Y/N | Metrik | Catatan |
| --- | --- | --- | --- |
| Ketersediaan tinggi | Y | 99,9% | |
| RTO | Y | 1 jam | Tujuan waktu pemulihan |
| RPO | Y | 1 jam | Tujuan titik pemulihan |
| Pemulihan bencana | N | | |
| Replikasi data di Wilayah | N | | |
| Replikasi data lintas wilayah | N | Latensi 3 detik | Yang mana Wilayah AWS? |

Tentukan persyaratan keamanan:

| Persyaratan | Y/N | Catatan |
| --- | --- | --- |
| Penyimpanan data sensitif | N | Informasi kesehatan yang dilindungi (PHI), informasi |

| | | |
|---|---|---|
| | | industri kartu pembayaran (PCI), informasi identitas pribadi (PII)? |
| Enkripsi saat istirahat | Y | |
| Enkripsi dalam perjalanan | Y | |
| Enkripsi sisi klien | N | |
| Pustaka enkripsi proprietary atau vendor ketiga | N | |
| Pencatatan akses data | N | |
| Audit akses data | N | |

## Templat pola akses

Kumpulkan dan dokumentasikan informasi tentang pola akses untuk kasus penggunaan dengan menggunakan bidang berikut:

| Bidang | Deskripsi |
|---|---|
| Pola akses | Berikan nama untuk pola akses. |
| Deskripsi | Berikan deskripsi yang lebih rinci tentang pola akses. |
| Prioritas | Tentukan prioritas untuk pola akses (tinggi, sedang, atau rendah). Ini mendefinisikan pola akses yang paling relevan untuk aplikasi. |
| Membaca atau menulis | Apakah ini pola akses baca atau akses tulis? |
| Jenis | Apakah pola mengakses satu item, beberapa item, atau semua item? |
| Filter | Apakah pola akses memerlukan filter? |

| Bidang | Deskripsi |
| --- | --- |
| Urutkan | Apakah hasilnya memerlukan penyortiran? |

## Templat

| Pola akses | Deskripsi | Prioritas | Membaca atau menulis | Jenis (item tunggal, beberapa item, atau semua) | Atribut kunci | Filter | Pemesanan hasil |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Buat profil pengguna | Pengguna membuat profil baru. | Tinggi | Menulis | Item tunggal | Nama Pengguna | N/A | N/A |
| Perbarui profil pengguna | Pengguna memperbarui profil mereka. | Sedang | Menulis | Item tunggal | Nama Pengguna | Nama pengguna = pengguna saat ini | N/A |
| Dapatkan profil pengguna | Pengguna meninjau profil mereka. | Tinggi | Baca | Item tunggal | Nama Pengguna | Nama pengguna = pengguna saat ini | N/A |
| Buat game | Pengguna membuat game baru. | Tinggi | Menulis | Item tunggal | GameID | N/A | N/A |

| | | | | | | |
|---|---|---|---|---|---|---|
| Temukan game terbuka | Pengguna mencari game terbuka. Hasil pencarian diurutkan berdasarkan stempel waktu mulai dalam urutan menurun. | Tinggi | Baca | Beberapa item | GameStatus = terbuka | Mulai stempel waktu turun |
| Temukan game terbuka berdasarkan peta | Pengguna mencari game terbuka dengan menggunakan peta tertentu yang diurutkan berdasarkan stempel waktu mulai saat turun pesanan. | Sedang | Baca | Beberapa item | GameStatus = terbuka dan Peta = XYZ | Mulai stempel waktu turun |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Lihat permainan | Pengguna meninjau detail permainan. | Tinggi | Baca | Item tunggal | GameID | N/A | N/A |
| Lihat pengguna dalam game | Pengguna mendapatkan daftar semua pengguna dalam sebuah game. | Sedang | Baca | Beberapa item | | GameID = XYZ | N/A |
| Bergabunglah dengan pengguna ke game | Pengguna bergabung dengan game terbuka. | Tinggi | Menulis | Item tunggal | GameID dan Nama Pengguna | GameStatus = terbuka | N/A |
| Memulai permainan | Pengguna memulai permainan baru. | Tinggi | Menulis | Item tunggal | GameID | N/A | N/A |
| Perbarui game untuk pengguna | Perbarui posisi pengguna dalam game. | Sedang | Menulis | Item tunggal | GameID dan Nama Pengguna | N/A | N/A |
| Perbarui game | Game berakhir; perbarui statistik. | Sedang | Menulis | Item tunggal | GameID | N/A | N/A |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Temukan semua game sebelumnya untuk pengguna | Buat daftar semua game yang dimainkan pengguna yang diurutkan berdasarkan stempel waktu awal permainan. | Rendah | Baca | Beberapa item | Nama pengguna dan GameID | Nama pengguna = pengguna saat ini | Mulai stempel waktu |
| Ekspor data untuk analitik data | Tim pengembangan akan menjalankan pekerjaan batch untuk mengekspor data ke Amazon S3. | Rendah | Baca | Semua | N/A | N/A | N/A |

# Praktik terbaik

Pertimbangkan untuk menggunakan praktik terbaik desain DynamoDB berikut:

- Desain kunci partisi - Gunakan kunci partisi kardinalitas tinggi untuk mendistribusikan beban secara merata.
- Pola desain daftar kedekatan - Gunakan pola desain ini untuk mengelola one-to-many dan many-to-many hubungan.
- Indeks jarang — Gunakan indeks jarang untuk indeks sekunder global Anda (). GSIs Saat Anda membuat GSI, Anda menentukan kunci partisi dan secara opsional kunci pengurutan. Hanya item di tabel dasar yang berisi kunci partisi GSI yang sesuai yang muncul di indeks jarang. Ini membantu untuk tetap GSIs lebih kecil.
- Index overloading — Gunakan GSI yang sama untuk mengindeks berbagai jenis item.
- GSI menulis sharding — Shard dengan bijak untuk mendistribusikan data di seluruh partisi untuk kueri yang efisien dan lebih cepat.
- Item besar — Simpan hanya metadata di dalam tabel, simpan gumpalan di Amazon S3, dan simpan referensi di DynamoDB. Pecahkan item besar menjadi beberapa item, dan indeks secara efisien dengan menggunakan kunci pengurutan.

Untuk praktik terbaik desain lainnya, lihat dokumentasi Amazon DynamoDB.

# Contoh pemodelan data hierarkis

Bagian berikut menggunakan contoh perusahaan otomotif untuk menunjukkan bagaimana Anda dapat menggunakan langkah-langkah proses pemodelan data untuk merancang sistem manajemen komponen multi-level di DynamoDB.

Topik



## Langkah 1: Identifikasi kasus penggunaan dan model data logis

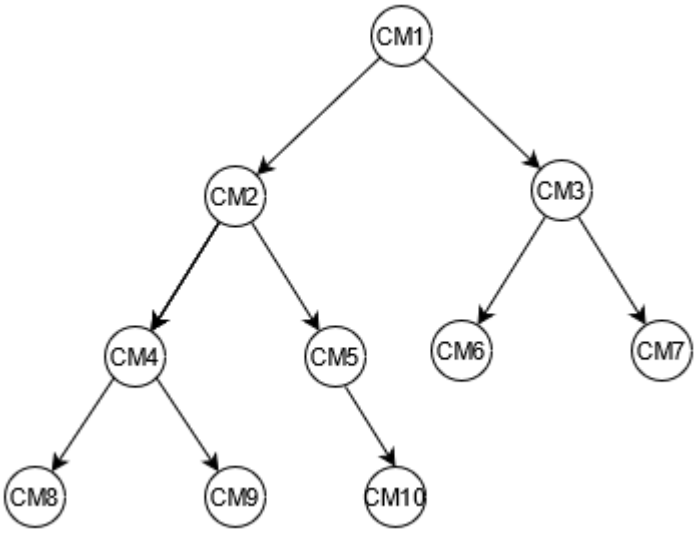
Sebuah perusahaan otomotif ingin membangun sistem manajemen komponen transaksional untuk menyimpan dan mencari semua suku cadang mobil yang tersedia dan untuk membangun hubungan antara komponen dan suku cadang yang berbeda. Misalnya, mobil berisi banyak baterai, setiap baterai berisi beberapa modul tingkat tinggi, setiap modul berisi banyak sel, dan setiap sel berisi beberapa komponen tingkat rendah.

Umumnya, untuk membangun model hubungan hierarkis, database grafik seperti Amazon Neptunus adalah pilihan yang lebih baik. Namun, dalam beberapa kasus, Amazon DynamoDB adalah alternatif yang lebih baik untuk pemodelan data hierarkis karena fleksibilitas, keamanan, kinerja, dan skalanya.

Misalnya, Anda mungkin membangun sistem di mana 80-90 persen kueri bersifat transaksional, di mana DynamoDB cocok. Dalam contoh ini, 10-20 persen kueri lainnya bersifat relasional, di mana database grafik seperti Neptunus lebih cocok. Dalam hal ini, termasuk database tambahan dalam arsitektur untuk memenuhi hanya 10-20 persen dari kueri dapat meningkatkan biaya. Ini juga

menambah beban operasional pemeliharaan beberapa sistem dan menyinkronkan data. Sebagai gantinya, Anda dapat memodelkan kueri relasional 10-20 persen di DynamoDB.

Membuat diagram pohon contoh untuk komponen mobil dapat membantu Anda memetakan hubungan di antara mereka. Diagram berikut menunjukkan grafik ketergantungan dengan empat level. CM1 adalah komponen tingkat atas untuk contoh mobil itu sendiri. Ini memiliki dua subkomponen untuk dua contoh baterai, CM2 dan CM3. Setiap baterai memiliki dua subkomponen, yaitu modul. CM2 memiliki modul CM4 dan CM5, dan CM3 memiliki modul CM6 dan CM7. Setiap modul memiliki beberapa subkomponen, yang merupakan sel. CM4 Modul ini memiliki dua sel, CM8 dan CM9. CM5 memiliki satu sel CM10. CM6 dan CM7 belum memiliki sel terkait.




Panduan ini akan menggunakan pohon ini dan pengidentifikasi komponennya sebagai referensi. Komponen teratas akan disebut sebagai induk, dan subkomponen akan disebut sebagai anak. Misalnya, komponen atas CM1 adalah induk dari CM2 dan CM3. CM2 adalah orang tua dari CM4 dan CM5. Ini menggambarkan hubungan orangtua-anak.

Dari pohon, Anda dapat melihat grafik ketergantungan lengkap komponen. Misalnya, CM8 tergantung pada CM4, yang tergantung pada CM2, yang bergantung pada CM1. Pohon mendefinisikan grafik ketergantungan lengkap sebagai jalur. Sebuah jalan menggambarkan dua hal:

- Grafik ketergantungan
- Posisi di pohon

Mengisi template untuk persyaratan bisnis:

Berikan informasi tentang pengguna Anda:

| Pengguna | Deskripsi |
| --- | --- |
| Karyawan | Karyawan internal perusahaan otomotif yang membutuhkan informasi mobil dan komponenn ya |

Berikan informasi tentang sumber data dan bagaimana data akan dicerna:

| Sumber | Deskripsi | Pengguna |
| --- | --- | --- |
| Sistem manajemen | Sistem yang akan menyimpan semua data yang terkait dengan suku cadang mobil yang tersedia dan hubungann ya dengan komponen dan suku cadang lainnya. | Karyawan |

Berikan informasi tentang bagaimana data akan dikonsumsi:

| Konsumen | Deskripsi | Pengguna |
| --- | --- | --- |
| Sistem manajemen | Ambil semua komponen turunan langsung untuk ID komponen induk. | Karyawan |
| Sistem manajemen | Ambil daftar rekursif semua komponen anak untuk ID komponen. | Karyawan |
| Sistem manajemen | Lihat nenek moyang suatu komponen. | Karyawan |

# Langkah 2: Buat estimasi biaya awal

Penting untuk menghitung estimasi biaya untuk semua lingkungan aplikasi Anda sehingga Anda dapat memeriksa apakah solusinya layak secara finansial. Praktik terbaik adalah membuat estimasi tingkat tinggi dan mendapatkan persetujuan dari analis bisnis sebelum melanjutkan pengembangan dan penyebaran.

- Insinyur database membuat analisis biaya awal menggunakan informasi yang tersedia dan contoh yang disajikan pada halaman harga DynamoDB.
  - Buat perkiraan biaya untuk kapasitas sesuai permintaan (lihat contoh).
  - Buat estimasi biaya untuk kapasitas yang disediakan (lihat contoh).
    - Untuk model kapasitas yang disediakan, dapatkan estimasi biaya dari kalkulator, dan terapkan diskon untuk kapasitas cadangan.
  - Bandingkan perkiraan biaya dari dua model kapasitas.
  - Buat estimasi untuk semua lingkungan (Dev, Prod, QA).
- Analis bisnis meninjau dan menyetujui atau menolak perkiraan biaya awal.

Dengan menggunakan nilai referensi ini, Anda dapat membuat perkiraan harga untuk dikirimkan untuk persetujuan. Untuk membuat anggaran, Anda dapat menggunakan halaman harga DynamoDB dan. AWS Kalkulator Harga

# Langkah 3: Identifikasi pola akses data Anda

Contoh kasus penggunaan ini memiliki pola akses berikut untuk mengelola hubungan antara komponen mobil yang berbeda.

| Pola akses | Prioritas | Membaca atau menulis | Deskripsi | Jenis | Filter | Pemesanan hasil |
|---|---|---|---|---|---|---|
| Anak langsung | Tinggi | Baca | Ambil semua komponen turunan langsung untuk ID | Beberapa | `Component ID` | N/A |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | komponen induk. | | | |
| Semua komponen anak | Tinggi | Baca | Ambil daftar rekursif semua komponen anak untuk ID komponen. | Beberapa | `Component ID` | N/A |
| Leluhur | Tinggi | Baca | Ambil nenek moyang suatu komponen. | Beberapa | `Component ID` | N/A |

## Langkah 4: Identifikasi persyaratan teknis

Contoh ini tidak memiliki persyaratan teknis khusus, yang berada di luar cakupan contoh ini. Dalam kasus nyata, ini adalah praktik terbaik untuk menyelesaikan langkah ini dan untuk memvalidasi bahwa semua persyaratan teknis terpenuhi sebelum melanjutkan dengan pengembangan dan penyebaran. Anda dapat menggunakan contoh kuesioner untuk menyelesaikan langkah ini dalam kasus bisnis Anda. Selain itu, kami sarankan untuk memvalidasi kuota layanan DynamoDB untuk memastikan bahwa tidak ada batasan keras dalam solusi yang Anda desain.

## Langkah 5: Buat model data DynamoDB

Tentukan kunci partisi untuk tabel dasar Anda dan indeks sekunder global (GSIs):

- Mengikuti praktik terbaik desain kunci, gunakan `ComponentId` sebagai kunci partisi untuk tabel dasar dalam contoh ini. Karena unik, `ComponentId` bisa menawarkan granularitas. DynamoDB menggunakan nilai hash dari kunci partisi Anda untuk menentukan partisi tempat data disimpan secara fisik. ID komponen unik menghasilkan nilai hash yang berbeda, yang dapat memfasilitasi distribusi data di dalam tabel. Anda dapat menanyakan tabel dasar dengan menggunakan kunci `ComponentId` partisi.

- Untuk menemukan turunan langsung dari komponen, buat GSI di mana `ParentId` kunci partisi, dan `ComponentId` merupakan kunci pengurutan. Anda dapat menanyakan GSI ini dengan menggunakan `ParentId` sebagai kunci partisi.
- Untuk menemukan semua turunan rekursif dari sebuah komponen, buat GSI di mana `GraphId` kunci partisi, dan `Path` merupakan kunci pengurutan. Anda dapat menanyakan GSI ini dengan menggunakan `GraphId` sebagai kunci partisi dan `BEGINS_WITH(Path, "$path")` operator pada tombol sortir.

| | Kunci partisi | Sortir Kunci | Atribut pemetaan |
|---|---|---|---|
| Tabel dasar | `ComponentId` | | `ParentId`, `GraphId`, `Path` |
| GSI1 | `ParentId` | `ComponentId` | |
| GSI2 | `GraphId` | `Path` | `ComponentId` |

## Menyimpan komponen dalam tabel

Langkah selanjutnya adalah menyimpan setiap komponen dalam tabel dasar DynamoDB. Setelah Anda memasukkan semua komponen dari pohon contoh, Anda mendapatkan tabel dasar berikut.

| ComponentId | ParentId | GraphId | Jalan |
|---|---|---|---|
| CM1 | | CM1#1 | CM1 |
| CM2 | CM1 | CM1#1 | CM1\|CM2 |
| CM3 | CM1 | CM1#1 | CM1\|CM3 |

| | | | |
|---|---|---|---|
| CM4 | CM2 | CM1#1 | CM1\|CM2\|CM4 |
| CM5 | CM2 | CM1#1 | CM1\|CM2\|CM5 |
| CM6 | CM3 | CM1#1 | CM1\|CM3\|CM6 |
| CM7 | CM3 | CM1#1 | CM1\|CM3\|CM7 |
| CM8 | CM4 | CM1#1 | CM1\|CM2\|CM4\|CM8 |
| CM9 | CM4 | CM1#1 | CM1\|CM2\|CM4\|CM9 |
| CM10 | CM5 | CM1#1 | CM1\|CM2\|CM5\|CM10 |

## GSI1 Indeks

Untuk memeriksa semua turunan langsung dari komponen, Anda membuat indeks yang digunakan `ParentId` sebagai kunci partisi dan `ComponentId` sebagai kunci pengurutan. Tabel pivot berikut mewakili indeks. GSI1 Anda dapat menggunakan indeks ini untuk mengambil semua komponen turunan langsung dengan menggunakan ID komponen induk. Misalnya, Anda dapat mengetahui berapa banyak baterai yang tersedia di mobil (CM1) atau sel mana yang tersedia dalam modul (CM4).

| ParentId | ComponentId |
|---|---|
| CM1 | CM2 |
| | CM3 |

| | |
|---|---|
| CM2 | CM4 |
| | CM5 |
| CM3 | CM6 |
| | CM7 |
| CM4 | CM8 |
| | CM9 |
| CM5 | CM10 |

## GSI2 Indeks

Tabel pivot berikut mewakili indeks. GSI2 Ini dikonfigurasi menggunakan `GraphId` sebagai kunci partisi dan `Path` sebagai kunci pengurutan. Menggunakan `GraphI` d dan `begins_with` operasi pada kunci sortir (`Path`), Anda dapat menemukan garis keturunan lengkap komponen dalam pohon.

| GraphId | Jalan | ComponentId |
|---|---|---|
| CM1#1 | CM1 | CM1 |
| | CM1\|CM2 | CM2 |
| | CM1\|CM3 | CM3 |
| | CM1\|CM2\|CM4 | CM4 |
| | CM1\|CM2\|CM5 | CM5 |
| | CM1\|CM2\|CM4\|CM8 | CM8 |
| | CM1\|CM2\|CM4\|CM9 | CM9 |
| | CM1\|CM2\|CM5\|CM10 | CM10 |
| | CM1\|CM3\|CM6 | CM6 |

| | |
|---|---|
| CM1\|CM3\|CM7 | CM7 |

# Langkah 6: Buat kueri data

Setelah Anda menentukan pola akses dan mendesain model data Anda, Anda dapat melakukan kueri data hierarkis dalam database DynamoDB. Sebagai praktik terbaik untuk menghemat biaya dan membantu memastikan kinerja, contoh berikut hanya menggunakan operasi kueri tanpa`Scan`.

- Temukan nenek moyang suatu komponen.

  Untuk menemukan leluhur (induk, kakek-nenek, kakek buyut, dan sebagainya) dari CM8 komponen, kueri tabel dasar menggunakan. `ComponentId = "CM8"` Kueri akan mengembalikan catatan berikut.

  Untuk mengurangi ukuran data hasil, Anda dapat menggunakan ekspresi proyeksi untuk mengembalikan hanya `Path` atribut.

  | ComponentId | ParentId | GraphId | Jalan |
  |---|---|---|---|
  | CM8 | CM4 | CM1#1 | CM1\|CM2\|CM4\|CM8 |

  | Jalan |
  |---|
  | CM1\|CM2\|CM4\|CM8 |

  Sekarang, pisahkan jalur menggunakan pipa (“|”), dan ambil komponen N-1 pertama untuk mendapatkan leluhur.

  Hasil kueri: Nenek moyang CM8 adalah CM1, CM2, CM4.

- Temukan anak-anak langsung dari suatu komponen.

  Untuk mendapatkan semua turunan langsung, atau satu tingkat hilir, komponen untuk komponen, kueri GSI1 menggunakan. CM2 `ParentId = "CM2"` Kueri akan mengembalikan catatan berikut.

  | ParentId | ComponentId |
  |---|---|

| | |
|---|---|
| CM2 | CM4 |
| | CM5 |

- Temukan semua komponen turunan hilir menggunakan komponen tingkat atas.

  Untuk mendapatkan semua komponen turunan, atau hilir, untuk komponen tingkat atas CM1, GSI2 menggunakan kueri `GraphId = "CM1#1"` dan`begins_with("Path", "CM1|")`, dan gunakan ekspresi proyeksi dengan. `ComponentId` Ini akan mengembalikan semua komponen yang terkait dengan pohon itu.

  Contoh ini memiliki satu pohon, dengan CM1 sebagai komponen teratas. Pada kenyataannya, Anda bisa memiliki jutaan komponen tingkat atas dalam tabel yang sama.

| GraphId | ComponentId |
|---|---|
| | CM2 |
| CM1#1 | CM3 |
| | CM4 |
| | CM5 |
| | CM8 |
| | CM9 |
| | CM10 |
| | CM6 |
| | CM7 |

- Temukan semua komponen turunan hilir menggunakan komponen tingkat menengah.

  Untuk mendapatkan semua komponen turunan, atau hilir, secara rekursif untuk komponen CM2, Anda memiliki dua opsi. Anda dapat menanyakan tingkat demi tingkat secara rekursif, atau Anda dapat menanyakan GSI2 indeks.

  - Kueri GSI1, tingkat demi level, secara rekursif, hingga mencapai tingkat terakhir komponen anak.
    1. Kueri GSI1 menggunakan`ParentId = "CM2"`. Ini akan mengembalikan catatan berikut.

| ParentId | ComponentId |
| --- | --- |
| CM2 | CM4 |
| | CM5 |

2. Sekali lagi, kueri GSI1 menggunakan`ParentId = "CM4"`. Ini akan mengembalikan catatan berikut.

| ParentId | ComponentId |
| --- | --- |
| CM4 | CM8 |
| | CM9 |

3. Sekali lagi, kueri GSI1 menggunakan`ParentId = "CM5"`. Ini akan mengembalikan catatan berikut.

   Lanjutkan loop: Kueri untuk masing-masing `ComponentId` sampai Anda mencapai level terakhir. Ketika kueri yang menggunakan `ParentId = "<ComponentId>"` tidak mengembalikan hasil apa pun, hasil sebelumnya berasal dari tingkat terakhir pohon.

| ParentId | ComponentId |
| --- | --- |
| CM5 | CM10 |

4. Gabungkan semua hasil.

   hasil= [CM4, CM5] + [CM8, CM9] + [] CM10

   =[CM4, CM5, CM8, CM9, CM10]

- Query GSI2, yang menyimpan pohon hierarkis untuk komponen tingkat atas (mobil, atau CM1).
  1. Pertama, temukan komponen tingkat atas atau leluhur teratas dan `Path` dari. CM2 Untuk itu, kueri tabel dasar dengan menggunakan `ComponentId = "CM2"` untuk menemukan jalur komponen itu di pohon hierarkis. Pilih atribut `GraphId` dan `Pat h`. Kueri akan mengembalikan catatan berikut.

| GraphId | Jalan |
| --- | --- |
| CM1#1 | CM1\|CM2 |

2. Query GSI2 dengan menggunakan`GraphId = "CM1#1" AND BEGINS_WITH("Path", "CM1|CM2|")`. Kueri akan mengembalikan hasil berikut.

| GraphId | Jalan | ComponentId |
| --- | --- | --- |
| CM1#1 | CM1\|CM2\|CM4 | CM4 |
| | CM1\|CM2\|CM5 | CM5 |
| | CM1\|CM2\|CM4\|CM8 | CM8 |
| | CM1\|CM2\|CM4\|CM9 | CM9 |
| | CM1\|CM2\|CM5\|CM10 | CM10 |

3. Pilih `ComponentId` atribut untuk mengembalikan semua komponen anak CM2.

# Langkah 7: Validasi model data

Pada langkah ini, pengguna bisnis memvalidasi hasil kueri dan memeriksa apakah mereka memenuhi kebutuhan bisnis. Anda dapat menggunakan tabel berikut untuk memeriksa pola akses terhadap persyaratan pengguna.

| Pertanyaan | Tabel dasar/GSI | Kueri |
| --- | --- | --- |
| Sebagai pengguna, saya ingin mengambil semua komponen turunan langsung untuk ID komponen induk. | GSI1 | `ParentId = "<ComponentId>"`<br>(Temukan anak-anak langsung dari suatu komponen.) |
| Sebagai pengguna, saya ingin mengambil daftar rekursif | GSI1 atau GSI2 | GSI1: `ParentId = "<ComponentId>"` |

| | | |
|---|---|---|
| semua komponen anak untuk ID komponen. | | atau<br><br>GSI2: `GraphId = "<TopLevelComponentId>#N" AND BEGINS_WITH("Path", "<PATH_OF_Component>")`<br><br>(Temukan semua komponen turunan tingkat bawah menggunakan komponen tingkat atas. Temukan semua komponen turunan tingkat bawah menggunakan komponen tingkat menengah.) |
| Sebagai pengguna, saya ingin melihat nenek moyang suatu komponen. | Tabel dasar | `ComponentId = "<ComponentId>"` , lalu pilih atribut Path.<br><br>(Temukan nenek moyang suatu komponen.) |

Anda juga dapat menerapkan skrip (tes) dalam bahasa pemrograman apa pun untuk menanyakan DynamoDB secara langsung dan membandingkan hasilnya dengan hasil yang diharapkan.

## Langkah 8: Tinjau estimasi biaya

Tinjau dan perbaiki estimasi biaya lagi. Selain itu, ini adalah praktik yang baik untuk memvalidasinya dengan pemangku kepentingan bisnis dan mendapatkan persetujuan untuk pindah ke langkah berikutnya.

### Tujuan

- Tentukan model kapasitas, dan perkirakan biaya DynamoDB untuk menyempurnakan estimasi biaya dari langkah 2.
- Dapatkan persetujuan keuangan akhir dari analis bisnis dan pemangku kepentingan.

## Proses

- Insinyur database mengidentifikasi estimasi volume data.
- Insinyur database mengidentifikasi persyaratan transfer data.
- Insinyur basis data mendefinisikan unit kapasitas baca dan tulis yang diperlukan.
- Analis bisnis memutuskan antara model kapasitas sesuai permintaan dan yang disediakan.
- Insinyur database mengidentifikasi kebutuhan untuk penskalaan otomatis DynamoDB.
- Insinyur basis data memasukkan parameter dalam file. AWS Kalkulator Harga
- Insinyur database menyajikan estimasi harga akhir kepada pemangku kepentingan bisnis.
- Analis bisnis dan pemangku kepentingan menyetujui atau menolak solusi.

# Langkah 9: Menyebarkan model data

Untuk contoh spesifik ini, penyebaran model dilakukan dengan menggunakan NoSQL Workbench, aplikasi untuk pengembangan dan operasi database modern. Dengan menggunakan alat ini, Anda memiliki opsi untuk membuat model data, mengunggah data, dan menerapkannya langsung ke Anda. Akun AWS Jika Anda ingin menerapkan contoh ini, Anda dapat menggunakan AWS CloudFormation template berikut, yang dihasilkan oleh NoSQL Workbench.

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  Components:
    Type: 'AWS::DynamoDB::Table'
    Properties:
      KeySchema:
        - AttributeName: ComponentId
          KeyType: HASH
      AttributeDefinitions:
        - AttributeName: ComponentId
          AttributeType: S
        - AttributeName: ParentId
          AttributeType: S
        - AttributeName: GraphId
          AttributeType: S
        - AttributeName: Path
          AttributeType: S
      GlobalSecondaryIndexes:
        - IndexName: GS1
```

```
        KeySchema:
          - AttributeName: ParentId
            KeyType: HASH
          - AttributeName: ComponentId
            KeyType: RANGE
        Projection:
          ProjectionType: KEYS_ONLY
      - IndexName: GSI2
        KeySchema:
          - AttributeName: GraphId
            KeyType: HASH
          - AttributeName: Path
            KeyType: RANGE
        Projection:
          ProjectionType: INCLUDE
          NonKeyAttributes:
            - ComponentId
    BillingMode: PAY_PER_REQUEST
    TableName: Components
```

# Sumber daya tambahan

Informasi lebih lanjut tentang DynamoDB

- Harga DynamoDB
- Dokumentasi DynamoDB
- Desain NoSQL untuk DynamoDB
- Tulis sharding
- Indeks sekunder lokal () LSIs
- Indeks sekunder global () GSIs
- Kelebihan beban GSIs
- Sharding GSI
- Menggunakan GSIs untuk membuat replika yang pada akhirnya konsisten
- Indeks jarang
- Kueri agregasi terwujud
- Pola desain deret waktu
- Pola desain daftar kedekatan
- Model kapasitas sesuai permintaan dan disediakan
- Penskalaan otomatis DynamoDB
- DynamoDB Waktu untuk Hidup (TTL)
- Memodelkan data pemain game dengan DynamoDB (lab)

AWS layanan

- AWS CloudFormation
- Amazon S3

Alat

- AWS Kalkulator Harga
- NoSQL Workbench untuk DynamoDB
- DynamoDB Lokal

- DynamoDB dan AWS SDKs

### Praktik terbaik

- Praktik terbaik untuk merancang dan merancang dengan DynamoDB (dokumentasi DynamoDB)
- Praktik terbaik untuk menggunakan indeks sekunder (dokumentasi DynamoDB)
- Praktik terbaik untuk menyimpan item dan atribut besar (dokumentasi DynamoDB)
- Memilih kunci AWS partisi DynamoDB yang tepat (Database blog)
- Cara mendesain indeks DBglobal sekunder Amazon Dynamo (blog AWS Database)
- Apa saja aspek di NoSQL Workbench untuk Amazon DynamoDB (situs web Medium)

### AWS sumber daya umum

- AWS Situs web Prescriptive Guidance
- AWS dokumentasi
- AWS referensi umum

# Kontributor

Kontributor untuk panduan ini meliputi:

- Camilo Gonzalez, Arsitek Data Senior, AWS
- Moinul Al-Mamun, Arsitek Data Besar Senior, AWS
- Santiago Segura, Konsultan Layanan Profesional, AWS
- Satheish Kumar Chandraprakasam, Arsitek Aplikasi Cloud, AWS

# 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 |
| --- | --- | --- |
| Menambahkan bagian Praktik terbaik dan contoh untuk pemodelan data hierarkis. | Kami menambahkan ringkasan praktik terbaik DynamoDB dan contoh merancang dan step-by-step memvalidasi model hierarkis. | 5 Desember 2023 |
| Publikasi awal | — | 26 Oktober 2020 |

# 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 AWS Region 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.

Pusat Keunggulan Cloud (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 CCo E 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 CCo E, 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. Bitbucket Cloud 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, Amazon SageMaker AI menyediakan algoritma 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 biasanya 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 di dalamnya 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 yang ada. 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 basis data.

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.

EDI

Lihat pertukaran data elektronik.

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.

pertukaran data elektronik (EDI)

Pertukaran otomatis dokumen bisnis antar organisasi. Untuk informasi selengkapnya, lihat Apa itu Pertukaran Data Elektronik.

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 dapat diakses oleh pengguna akhir. Dalam sebuah CI/CD pipeline, 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, AWS Region, 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.

beberapa tembakan mendorong

Menyediakan LLM dengan sejumlah kecil contoh yang menunjukkan tugas dan output yang diinginkan sebelum memintanya untuk melakukan tugas serupa. Teknik ini adalah aplikasi pembelajaran dalam konteks, di mana model belajar dari contoh (bidikan) yang tertanam dalam petunjuk. Beberapa bidikan dapat efektif untuk tugas-tugas yang memerlukan pemformatan, penalaran, atau pengetahuan domain tertentu. Lihat juga bidikan nol.

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.

FM

Lihat model pondasi.

model pondasi (FM)

Jaringan saraf pembelajaran mendalam yang besar yang telah melatih kumpulan data besar-besaran data umum dan tidak berlabel. FMs mampu melakukan berbagai tugas umum, seperti memahami bahasa, menghasilkan teks dan gambar, dan berbicara dalam bahasa alami. Untuk informasi selengkapnya, lihat Apa itu Model Foundation.

# G

AI generatif

Subset model AI yang telah dilatih pada sejumlah besar data dan yang dapat menggunakan prompt teks sederhana untuk membuat konten dan artefak baru, seperti gambar, video, teks, dan audio. Untuk informasi lebih lanjut, lihat Apa itu AI Generatif.

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.

gambar emas

Sebuah snapshot dari sistem atau perangkat lunak yang digunakan sebagai template untuk menyebarkan instance baru dari sistem atau perangkat lunak itu. Misalnya, di bidang manufaktur, gambar emas dapat digunakan untuk menyediakan perangkat lunak pada beberapa perangkat dan membantu meningkatkan kecepatan, skalabilitas, dan produktivitas dalam operasi manufaktur perangkat.

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 ()OUs. 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 CSPM, 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.

data penahanan

Sebagian dari data historis berlabel yang ditahan dari kumpulan data yang digunakan untuk melatih model pembelajaran mesin. Anda dapat menggunakan data penahanan untuk mengevaluasi kinerja model dengan membandingkan prediksi model dengan data penahanan.

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 inbound, outbound, dan inspeksi VPCs 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 lebih lanjut, 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 VPCs (dalam yang sama atau berbeda Wilayah AWS), internet, dan jaringan lokal. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan Anda dengan inbound, outbound, dan inspeksi VPCs 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 lebih lanjut, 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.

model bahasa besar (LLM)

Model AI pembelajaran mendalam yang dilatih sebelumnya pada sejumlah besar data. LLM dapat melakukan beberapa tugas, seperti menjawab pertanyaan, meringkas dokumen, menerjemahkan teks ke dalam bahasa lain, dan menyelesaikan kalimat. Untuk informasi lebih lanjut, lihat Apa itu LLMs.

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.

LLM

Lihat model bahasa besar.

lingkungan yang lebih rendah

Lihat lingkungan.

# M

pembelajaran mesin (ML)

Jenis kecerdasan buatan yang menggunakan algoritma dan teknik untuk pengenalan dan pembelajaran pola. 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 dengan jelas APIs 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 ringan. APIs 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 dan pelajaran terbaik 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. 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 inbound, outbound, dan inspeksi VPCs 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 kebijakan berbasis sumber daya), atau menentukan izin maksimum untuk semua akun di organisasi (lihat kebijakan kontrol layanan). AWS Organizations

ketekunan 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.

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 berdasarkan desain

Pendekatan rekayasa sistem yang memperhitungkan privasi melalui seluruh proses pengembangan.

zona yang dihosting pribadi

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

rantai cepat

Menggunakan output dari satu prompt LLM sebagai input untuk prompt berikutnya untuk menghasilkan respons yang lebih baik. Teknik ini digunakan untuk memecah tugas yang kompleks menjadi subtugas, atau untuk secara iteratif memperbaiki atau memperluas respons awal. Ini membantu meningkatkan akurasi dan relevansi respons model dan memungkinkan hasil yang lebih terperinci dan dipersonalisasi.

pseudonimisasi

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

publish/subscribe (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).

LAP

Lihat Retrieval Augmented Generation.

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 kehilangan 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.

Region

Kumpulan AWS sumber daya di wilayah geografis. Masing-masing AWS Region 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 prinsipal 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). Tipe 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.

Retrieval Augmented Generation (RAG)

Teknologi AI generatif di mana LLM merujuk sumber data otoritatif yang berada di luar sumber data pelatihannya sebelum menghasilkan respons. Misalnya, model RAG mungkin melakukan

pencarian semantik dari basis pengetahuan organisasi atau data kustom. Untuk informasi lebih lanjut, lihat Apa itu RAG.

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 Konsol Manajemen AWS 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.

SCADA

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.

keamanan dengan desain

Pendekatan rekayasa sistem yang memperhitungkan keamanan melalui seluruh proses pengembangan.

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 SCPs menentukan pagar pembatas atau menetapkan batasan pada tindakan yang dapat didelegasikan oleh administrator kepada pengguna atau peran. Anda dapat menggunakan SCPs 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.

sistem prompt

Teknik untuk memberikan konteks, instruksi, atau pedoman ke LLM untuk mengarahkan perilakunya. Permintaan sistem membantu mengatur konteks dan menetapkan aturan untuk interaksi dengan pengguna.

# T

tag

Pasangan nilai kunci yang bertindak sebagai metadata untuk mengatur sumber daya Anda. AWS Tanda 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 jaringan Anda VPCs dan lokal. 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 VPCs yang memungkinkan Anda untuk 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 AWS Kerangka Kualifikasi Beban Kerja.

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.

bisikan zero-shot

Memberikan LLM dengan instruksi untuk melakukan tugas tetapi tidak ada contoh (tembakan) yang dapat membantu membimbingnya. LLM harus menggunakan pengetahuan pra-terlatih untuk menangani tugas. Efektivitas bidikan nol tergantung pada kompleksitas tugas dan kualitas prompt. Lihat juga beberapa bidikan yang diminta.

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.