



Otimizando o desempenho de consultas do PostgreSQL

AWS Orientação prescritiva



AWS Orientação prescritiva: Otimizando o desempenho de consultas do PostgreSQL

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

As marcas comerciais e imagens de marcas da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestige a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

Table of Contents

Introdução	1
Casos de uso para ajuste de desempenho de consultas	1
PLANO EXPLIQUE	2
A declaração EXPLAIN	2
Usando EXPLAIN ANALYZE	2
Como ler o plano de consulta EXPLAIN	2
.....	4
Agrupamentos	13
Incompatibilidade de tipos de dados	16
Chamada de função em SELECT	18
EM ou EXISTE	19
Subconsultas ou CTEs	23
Perguntas frequentes	26
O que é EXPLAIN?	26
O que é EXPLAIN ANALYZE?	26
O que é agrupamento no PostgreSQL?	27
O que é um CTE?	27
Quais são as categorias de funções no PostgreSQL?	27
.....	29
Colaboradores	30
Histórico do documentos	31
Glossário	32
#	32
A	33
B	36
C	38
D	41
E	45
F	47
G	49
H	50
eu	52
L	54
M	55

O	60
P	62
Q	65
R	66
S	69
T	73
U	74
V	75
W	75
Z	76
.....	lxxviii

Otimizando o desempenho de consultas do PostgreSQL

Amazon Web Services ([colaboradores](#))

Abril de 2024 ([histórico do documento](#))

O PostgreSQL é um sistema de banco de dados objeto-relacional de código aberto que é poderoso, flexível e confiável. Há muitas maneiras de otimizar o desempenho de uma consulta do PostgreSQL. O processo de otimização da consulta depende do caso de uso. Conhecer o plano de consulta atual pode ajudá-lo a identificar e entender quaisquer problemas e fazer as alterações necessárias. Às vezes, talvez seja necessário analisar as tabelas para manter as estatísticas do banco de dados atualizadas. O otimizador do PostgreSQL usará essas estatísticas para executar a consulta mais rapidamente. Este guia se concentra nas melhores práticas para melhorar o desempenho das consultas do PostgreSQL.

Este guia pressupõe que você já tenha uma instância de banco de dados compatível com Amazon Relational Database Service (Amazon RDS) para PostgreSQL ou Amazon Aurora PostgreSQL.

Casos de uso para ajuste de desempenho de consultas

Este guia abrange cinco casos de uso, com explicações e exemplos:

- Agrupamentos
- Incompatibilidade de tipos de dados
- Chamada de função na SELECT declaração
- IN ou EXISTS
- Subconsultas ou expressões de tabela comuns (CTEs)

Cada caso de uso fornece detalhes do plano de execução inicial, como analisar o plano para identificar o problema e uma solução. A implementação desses casos de uso geralmente resulta em tempos de resposta mais rápidos para consultas, redução da carga no servidor e maior eficiência geral do sistema. Essas melhorias podem levar a uma melhor experiência do usuário e a uma maior confiabilidade do sistema.

O plano de consulta EXPLAIN

O PostgreSQL fornece EXPLAIN as opções EXPLAIN ANALYZE e para retornar planos de consulta com detalhes sobre como a consulta será executada.

A declaração EXPLAIN

A EXPLAIN instrução retorna o plano de consulta que o planejador do PostgreSQL gera para uma determinada instrução. O plano de consulta mostra o seguinte:

- Como as tabelas envolvidas em uma declaração serão digitalizadas (por exemplo, por varredura de índice ou varredura sequencial)
- Como várias tabelas serão unidas (por exemplo, junção de hash, junção de mesclagem ou junção de loop aninhada)

Compreender o plano é fundamental para melhorar o desempenho da consulta. Depois de entender o plano, você pode se concentrar em onde a consulta está demorando muito e tomar medidas para reduzir o tempo.

Usando EXPLAIN ANALYZE

No PostgreSQL EXPLAIN, só gerará um plano para a instrução dada. Se você adicionar a ANALYZE palavra-chave, EXPLAIN retornará o plano, executará a consulta e mostrará o tempo de execução real e a contagem de linhas para cada etapa. Isso é indispensável para analisar o desempenho da consulta.

Important

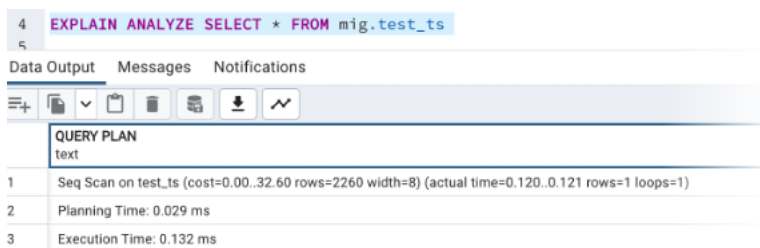
Ao usar EXPLAIN ANALYZE, tenha cuidado com INSERT UPDATE, DELETE e.

Como ler o plano de consulta EXPLAIN

Um plano de consulta do PostgreSQL é uma estrutura em árvore que consiste em vários nós. O plano de EXPLAIN consulta mostra as etapas que o mecanismo de banco de dados usa para executar uma consulta. O plano de consulta fornece as seguintes informações:

- O tipo de operação realizada, como varreduras sequenciais, varreduras de índice ou junções de loops aninhados.
- Um rótulo, como `Seq Scan`, ou `Index ScanNested Loop`, para descrever a operação que está sendo executada.
- O nome da tabela ou índice que está sendo processado pela consulta.
- Colunas de custo e linha com informações sobre o custo estimado em uma unidade arbitrária de computação e o número de linhas processadas.
- A condição do filtro de qualquer filtro aplicado na operação, como a `where` condição.
- Uma representação visual das etapas, com cada operação mostrada como um nó e setas conectando as operações. A ordem das operações é mostrada da esquerda para a direita, com as operações anteriores alimentando as operações posteriores.

A captura de tela a seguir mostra o plano de consulta para uma varredura sequencial.



The screenshot shows a PostgreSQL query planner output for the query `EXPLAIN ANALYZE SELECT * FROM mig.test_ts`. The output is displayed in a table with three rows. The first row shows the operation `Seq Scan on test_ts` with estimated cost `(cost=0.00..32.60 rows=2260 width=8)` and actual execution time `(actual time=0.120..0.121 rows=1 loops=1)`. The second row shows the planning time `Planning Time: 0.029 ms`. The third row shows the execution time `Execution Time: 0.132 ms`.

	QUERY PLAN
1	Seq Scan on test_ts (cost=0.00..32.60 rows=2260 width=8) (actual time=0.120..0.121 rows=1 loops=1)
2	Planning Time: 0.029 ms
3	Execution Time: 0.132 ms

A estimativa de custo (`cost=0.00..32.60 rows=2260 width=8`) significa que o PostgreSQL espera que a consulta exija 32,60 unidades de computação para retornar os resultados.

O `0.00` valor é o custo pelo qual esse nó pode começar a trabalhar (nesse caso, o tempo de inicialização da consulta). O `rows` valor é o número estimado de linhas que a varredura sequencial retornará. O `width` valor é o tamanho estimado em bytes das linhas retornadas.

Como o exemplo mostra `EXPLAIN` com a `ANALYZE` opção, a consulta foi executada e as informações de tempo foram capturadas. O resultado (`actual time=0.120..0.121 rows=1 loops=1`) significa o seguinte:

- A varredura sequencial foi executada uma vez (o `loops` valor).
- A digitalização retornou uma linha.
- O tempo real foi de 0,12 milissegundos.

Casos de uso para ajustar consultas

Este guia aborda os seguintes casos de uso para ajustar o desempenho da consulta:

- Agrupamentos
- Incompatibilidade de tipos de dados
- Chamada de função na SELECT declaração
- IN ou EXISTS
- Subconsultas ou expressões de tabela comuns (CTEs)

Para testar o ajuste de desempenho para esses casos de uso de desempenho de consulta, use seu banco de dados existente e os dados de exemplo fornecidos por este guia. O exemplo usa dados de uma companhia aérea XX fictícia. Para preparar os dados de exemplo, execute o seguinte código de exemplo:

```
--Creating required tables along with data.

--Creating user and schema
create user perf_user;
create schema perf_user AUTHORIZATION perf_user;
set search_path to perf_user;

--Table1:

CREATE TABLE IF NOT EXISTS perf_user.rnr_expiry_date
(
    airline_iata_code character(2) COLLATE pg_catalog."default",
    pnr_number character varying(15) COLLATE pg_catalog."default" NOT NULL,
    calculated_pnr_expiry_date timestamp(0) without time zone,
    row_num bigint,
    arc_expiry_date timestamp(0) without time zone,
    status character varying(10) COLLATE pg_catalog."default"
);

insert into perf_user.rnr_expiry_date
select 'XX' , upper(substring(concat(md5(random()::text), md5(random()::text)), 0,
7)), '2023-01-01 00:00:00', generate_series(1,100000), '2023-02-02 00:00:00' ,null;
```

```

CREATE INDEX rnr_expiry_date_idx1 ON perf_user.rnr_expiry_date (row_num ASC NULLS
LAST);

CREATE INDEX rnr_expiry_date_idx2 ON perf_user.rnr_expiry_date (airline_iata_code
COLLATE pg_catalog."default" ASC NULLS LAST, pnr_number COLLATE pg_catalog."default"
ASC NULLS LAST);

CREATE INDEX rnr_expiry_date_idx3 ON perf_user.rnr_expiry_date (pnr_number ASC NULLS
LAST);

vacuum analyze perf_user.rnr_expiry_date;

-----
--Table2:

CREATE TABLE IF NOT EXISTS perf_user.rnr_segment_pax
(
    airline_iata_code character varying(6) COLLATE pg_catalog."default" NOT NULL,
    pnr_number character varying(15) COLLATE pg_catalog."default" NOT NULL,
    segment_pax_id numeric(25,0) NOT NULL,
    oandd_id numeric(25,0) NOT NULL,
    segment_id numeric(25,0) NOT NULL,
    cabin_class character varying(15) COLLATE pg_catalog."default",
    pax_id numeric(25,0) NOT NULL,
    ticket_number character varying(25) COLLATE pg_catalog."default",
    ticket_type character varying(10) COLLATE pg_catalog."default",
    archive_status smallint NOT NULL DEFAULT (0)::smallint,
    certificate_number character varying(100) COLLATE pg_catalog."default",
    loyalty_number character varying(25) COLLATE pg_catalog."default",
    arc_expiry_date timestamp(0) without time zone,
    CONSTRAINT rnr_segment_pax_pk PRIMARY KEY (airline_iata_code, pnr_number,
segment_id, pax_id),
    CONSTRAINT rnr_segment_pax_ck1 CHECK (ticket_type::text = ANY (ARRAY['E'::character
varying::text, 'A'::character varying::text, 'C'::character varying::text,
'M'::character varying::text, 'I'::character varying::text]))
);

insert into perf_user.rnr_segment_pax (airline_iata_code, pnr_number, segment_pax_id,
oandd_id, segment_id, pax_id, ticket_type, arc_expiry_date )
select 'XX',upper(substring(concat(md5(random()::text), md5(random()::text)), 0, 7)),
generate_series(1,10000000),generate_series(1,10000000),
generate_series(1,10000000),generate_series(1,10000000),'A','2023-01-01 00:00:00';

```

```
insert into perf_user.rnr_segment_pax (airline_iata_code, pnr_number, segment_pax_id,
  oandd_id, segment_id, pax_id, ticket_type, arc_expiry_date )
select 'XX',upper(substring(concat(md5(random())::text), md5(random())::text)), 0, 7)),
generate_series(10000001,20000000),generate_series(10000001,20000000),
generate_series(10000001,20000000),generate_series(10000001,20000000),'I','2023-01-01
00:00:00';
```

```
insert into perf_user.rnr_segment_pax (airline_iata_code, pnr_number, segment_pax_id,
  oandd_id, segment_id, pax_id, ticket_type, arc_expiry_date)
select 'XX',upper(substring(concat(md5(random())::text), md5(random())::text)), 0, 7)),
generate_series(20000001,30000000),generate_series(20000001,30000000),
generate_series(20000001,30000000),generate_series(20000001,30000000),'E','2023-01-01
00:00:00';
```

```
insert into perf_user.rnr_segment_pax (airline_iata_code, pnr_number, segment_pax_id,
  oandd_id, segment_id, pax_id, ticket_type, arc_expiry_date)
select 'XX',upper(substring(concat(md5(random())::text), md5(random())::text)), 0, 7)),
generate_series(30000001,40000000),generate_series(30000001,40000000),
generate_series(30000001,40000000),generate_series(30000001,40000000),'M','2023-01-01
00:00:00';
```

```
CREATE INDEX rnr_segment_pax_idx1
  ON perf_user.rnr_segment_pax USING btree
  (loyalty_number COLLATE pg_catalog."default" ASC NULLS LAST, airline_iata_code
  COLLATE pg_catalog."default" ASC NULLS LAST, arc_expiry_date ASC NULLS LAST);
```

```
CREATE INDEX IF NOT EXISTS rnr_segment_pax_pn_idx1
  ON perf_user.rnr_segment_pax USING btree
  (pnr_number COLLATE pg_catalog."default" ASC NULLS LAST);
```

```
CREATE INDEX IF NOT EXISTS rnr_segment_pax_seq_idx1
  ON perf_user.rnr_segment_pax USING btree
  (segment_id ASC NULLS LAST);
```

```
vacuum analyze perf_user.rnr_segment_pax;
```

```
-----
```

```
--Table3:
```

```
CREATE TABLE IF NOT EXISTS perf_user.rnr_segment
(
  airline_iata_code character varying(6) COLLATE pg_catalog."default" NOT NULL,
```

```

pnr_number character varying(15) COLLATE pg_catalog."C" NOT NULL,
segment_id numeric(25,0) NOT NULL,
oandd_id numeric(25,0),
price_id numeric(25,0),
flight_carrier character varying(6) COLLATE pg_catalog."default" ,
flight_number integer ,
flight_suffix character varying(1) COLLATE pg_catalog."default" ,
flight_date_ltc timestamp(0) without time zone ,
airline_company_code character varying(6) COLLATE pg_catalog."default",
bd_airport_code character varying(5) COLLATE pg_catalog."default" ,
off_airport_code character varying(5) COLLATE pg_catalog."default" ,
segment_status character varying(50) COLLATE pg_catalog."default" ,
flight_status character varying(30) COLLATE pg_catalog."default",
flight_type character varying(15) COLLATE pg_catalog."default",
cabin_class character varying(15) COLLATE pg_catalog."default",
arc_expiry_date timestamp(0) without time zone,
oandd_dep_date_ltc timestamp(0) without time zone,
added_time timestamp(6) without time zone,
dep_date_ltc timestamp(0) without time zone ,
arr_date_utc timestamp(0) without time zone,
dep_date_utc timestamp(0) without time zone,
origin character varying(5) COLLATE pg_catalog."default",
destination character varying(5) COLLATE pg_catalog."default",
CONSTRAINT rnr_segment_pk PRIMARY KEY (pnr_number, segment_id, airline_iata_code)
);

```

```

insert into perf_user.rnr_segment (airline_iata_code, pnr_number, segment_id,
  FLIGHT_CARRIER, FLIGHT_NUMBER, FLIGHT_SUFFIX, FLIGHT_DATE_LTC)
select 'XX',
upper(substring(concat(md5(random())::text), md5(random())::text)), 0, 7)),
generate_series(1,10000000),'XX',110,'*', '2023-01-01 00:00:00';

```

```

insert into perf_user.rnr_segment (airline_iata_code, pnr_number, segment_id,
  FLIGHT_CARRIER, FLIGHT_NUMBER, FLIGHT_SUFFIX, FLIGHT_DATE_LTC)
select 'XX',
upper(substring(concat(md5(random())::text), md5(random())::text)), 0, 7)),
generate_series(10000001,20000000),'XX',120,'*', '2023-01-01 00:00:00';

```

```

insert into perf_user.rnr_segment (airline_iata_code, pnr_number, segment_id,
  FLIGHT_CARRIER, FLIGHT_NUMBER, FLIGHT_SUFFIX, FLIGHT_DATE_LTC)
select 'XX',
upper(substring(concat(md5(random())::text), md5(random())::text)), 0, 7)),
generate_series(20000001,30000000),'XX',130,'*', '2023-01-01 00:00:00';

```

```

insert into perf_user.rnr_segment (airline_iata_code, pnr_number, segment_id,
    FLIGHT_CARRIER, FLIGHT_NUMBER, FLIGHT_SUFFIX, FLIGHT_DATE_LTC)
select 'XX',
upper(substring(concat(md5(random())::text), md5(random())::text), 0, 7)),
generate_series(30000001,40000000),'XX',140,'*', '2023-01-01 00:00:00';

insert into perf_user.rnr_segment (airline_iata_code, pnr_number, segment_id,
    FLIGHT_CARRIER, FLIGHT_NUMBER, FLIGHT_SUFFIX, FLIGHT_DATE_LTC)
select 'XX',
upper(substring(concat(md5(random())::text), md5(random())::text), 0, 7)),
generate_series(40000001,50000000),'XX',150,'*', '2023-01-01 00:00:00';

insert into perf_user.rnr_segment (airline_iata_code, pnr_number, segment_id,
    FLIGHT_CARRIER, FLIGHT_NUMBER, FLIGHT_SUFFIX, FLIGHT_DATE_LTC)
select 'XX',
upper(substring(concat(md5(random())::text), md5(random())::text), 0, 7)),
generate_series(50000001,60000000),'XX',160,'*', '2023-01-01 00:00:00';

CREATE INDEX rnr_segment_idx1 ON perf_user.rnr_segment USING btree
    (flight_date_ltc ASC NULLS LAST, bd_airport_code COLLATE pg_catalog."default"
    ASC NULLS LAST, off_airport_code COLLATE pg_catalog."default" ASC NULLS LAST,
    flight_number ASC NULLS LAST, flight_carrier COLLATE pg_catalog."default" ASC NULLS
    LAST, flight_suffix COLLATE pg_catalog."default" ASC NULLS LAST, airline_iata_code
    COLLATE pg_catalog."default" ASC NULLS LAST, arc_expiry_date ASC NULLS LAST);

CREATE INDEX rnr_segment_idx2
    ON perf_user.rnr_segment USING btree
    (dep_date_ltc ASC NULLS LAST, flight_number ASC NULLS LAST, bd_airport_code COLLATE
    pg_catalog."default" ASC NULLS LAST, off_airport_code COLLATE pg_catalog."default" ASC
    NULLS LAST, flight_carrier COLLATE pg_catalog."default" ASC NULLS LAST, flight_suffix
    COLLATE pg_catalog."default" ASC NULLS LAST, arc_expiry_date ASC NULLS LAST);

CREATE INDEX rnr_segment_idx3
    ON perf_user.rnr_segment USING btree
    (pnr_number COLLATE pg_catalog."default" ASC NULLS LAST, arr_date_utc ASC NULLS
    LAST, airline_iata_code COLLATE pg_catalog."default" ASC NULLS LAST, arc_expiry_date
    ASC NULLS LAST);

CREATE INDEX rnr_segment_idx4
    ON perf_user.rnr_segment USING btree
    (dep_date_utc ASC NULLS LAST, added_time ASC NULLS LAST, airline_iata_code COLLATE
    pg_catalog."default" ASC NULLS LAST, arc_expiry_date ASC NULLS LAST);

```

```

CREATE INDEX rnr_segment_idx5
    ON perf_user.rnr_segment USING btree
    (origin COLLATE pg_catalog."default" ASC NULLS LAST, destination COLLATE
pg_catalog."default" ASC NULLS LAST, oandd_dep_date_ltc ASC NULLS LAST,
airline_iata_code COLLATE pg_catalog."default" ASC NULLS LAST, arc_expiry_date ASC
NULLS LAST);

CREATE INDEX rnr_segment_idx6
    ON perf_user.rnr_segment USING btree
    (pnr_number COLLATE pg_catalog."default" ASC NULLS LAST, oandd_id ASC NULLS LAST,
segment_id ASC NULLS LAST, airline_iata_code COLLATE pg_catalog."default" ASC NULLS
LAST, arc_expiry_date ASC NULLS LAST);

vacuum analyze perf_user.rnr_segment;

-----

--Table4:

CREATE TABLE IF NOT EXISTS perf_user.rnr_seat_numbers
(
    airline_iata_code character varying(6) COLLATE pg_catalog."default" NOT NULL,
    pnr_number character varying(15) COLLATE pg_catalog."default" NOT NULL,
    segment_id numeric(25,0) NOT NULL,
    pax_id numeric(25,0) NOT NULL,
    seat_id numeric(25,0) NOT NULL,
    bd_airport_code character varying(5) COLLATE pg_catalog."default",
    off_airport_code character varying(5) COLLATE pg_catalog."default",
    seat_number character varying(5) COLLATE pg_catalog."default",
    seat_status character varying(20) COLLATE pg_catalog."default",
    ssr_id character varying(100) COLLATE pg_catalog."default",
    archive_status smallint DEFAULT (0)::smallint,
    seat_alloc_id numeric(25,0),
    archive_date timestamp(0) without time zone,
    seat_attribute_code character varying(201) COLLATE pg_catalog."default",
    channel_code character varying(20) COLLATE pg_catalog."default",
    arc_expiry_date timestamp(0) without time zone,
    CONSTRAINT rnr_seat_numbers_pk PRIMARY KEY (pnr_number, segment_id, pax_id,
seat_id, airline_iata_code)
);

```

```

insert into perf_user.rnr_seat_numbers (pnr_number, segment_id, pax_id, seat_id,
    airline_iata_code)
select upper(substring(concat(md5(random()::text), md5(random()::text)), 0, 7)),
generate_series(1,10000000),generate_series(1,10000000),generate_series(1,10000000),'XX';

insert into perf_user.rnr_seat_numbers (pnr_number, segment_id, pax_id, seat_id,
    airline_iata_code)
select upper(substring(concat(md5(random()::text), md5(random()::text)), 0, 7)),
generate_series(100000001,200000000),generate_series(100000001,200000000),generate_series(100000001,
200000000),'XX';

insert into perf_user.rnr_seat_numbers (pnr_number, segment_id, pax_id, seat_id,
    airline_iata_code)
select upper(substring(concat(md5(random()::text), md5(random()::text)), 0, 7)),
generate_series(200000001,300000000),generate_series(200000001,300000000),generate_series(200000001,
300000000),'XX';

insert into perf_user.rnr_seat_numbers (pnr_number, segment_id, pax_id, seat_id,
    airline_iata_code)
select upper(substring(concat(md5(random()::text), md5(random()::text)), 0, 7)),
generate_series(300000001,400000000),generate_series(300000001,400000000),generate_series(300000001,
400000000),'XX';

vacuum Analyze perf_user.rnr_seat_numbers;

--Table5:
CREATE TABLE IF NOT EXISTS perf_user.test_veh
(
    test_veh_id bigint NOT NULL,
    oiltype_id bigint,
    vehicle_id character varying(50) COLLATE pg_catalog."default",
    serviceprogram_id character varying(100) COLLATE pg_catalog."default",
    startdate timestamp without time zone,
    enddate timestamp without time zone,
    last_update_dt timestamp without time zone,
    CONSTRAINT test_veh_pkey PRIMARY KEY (test_veh_id),
    CONSTRAINT test_veh_oiltype_id_fkey FOREIGN KEY (oiltype_id)
        REFERENCES perf_user.oiltype (oiltype_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT test_veh_oiltype_id_fkey1 FOREIGN KEY (oiltype_id)
        REFERENCES perf_user.oiltype (oiltype_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
);

CREATE INDEX IF NOT EXISTS test_veh_enddate_ind

```

```
ON perf_user.test_veh USING btree
(enddate ASC NULLS LAST);

CREATE INDEX IF NOT EXISTS test_veh_oiltype_id_ind
ON perf_user.test_veh USING btree
(oiltype_id ASC NULLS LAST);

--Table6:
CREATE TABLE IF NOT EXISTS perf_user.oiltype
(
    oiltype_id bigint NOT NULL,
    descr character varying(50) COLLATE pg_catalog."default",
    CONSTRAINT oiltype_pkey PRIMARY KEY (oiltype_id)
);

CREATE INDEX IF NOT EXISTS oiltype_oiltyp_in
ON perf_user.oiltype USING btree
(oiltype_id ASC NULLS LAST);

--Table7:
CREATE TABLE IF NOT EXISTS perf_user.serviceprogram
(
    serial bigint NOT NULL,
    serviceprogram_id character varying(50) COLLATE pg_catalog."default",
    proname character varying(150) COLLATE pg_catalog."default",
    CONSTRAINT serviceprogram_pkey PRIMARY KEY (serial)
);

CREATE INDEX IF NOT EXISTS proname_id_ind
ON perf_user.serviceprogram USING btree
(proname COLLATE pg_catalog."default" ASC NULLS LAST);

CREATE INDEX IF NOT EXISTS serviceprogram_id_ind
ON perf_user.serviceprogram USING btree
(serviceprogram_id COLLATE pg_catalog."default" ASC NULLS LAST);

--Table8:
CREATE TABLE IF NOT EXISTS perf_user.vehicleservicehistory
(
    v_id bigint NOT NULL,
    test_veh_id bigint,
```

```

desc_1 character varying(50) COLLATE pg_catalog."default",
start_date timestamp without time zone,
end_date timestamp without time zone,
CONSTRAINT vehicleservicehistory_pkey PRIMARY KEY (v_id)
);

CREATE INDEX IF NOT EXISTS veh_end_date_id_ind
ON perf_user.vehicleservicehistory USING btree
(end_date ASC NULLS LAST);

CREATE INDEX IF NOT EXISTS veh_ser_ind
ON perf_user.vehicleservicehistory USING btree
(test_veh_id ASC NULLS LAST);

CREATE INDEX IF NOT EXISTS vehicleservicehistory_v_id_ind
ON perf_user.vehicleservicehistory USING btree
(test_veh_id ASC NULLS LAST);

--Function creation
CREATE OR REPLACE FUNCTION perf_user.return_data()
RETURNS character varying
LANGUAGE 'plpgsql'
COST 100
VOLATILE PARALLEL UNSAFE
AS $BODY$
BEGIN
return 'EE9F41' ;
END;
$BODY$;
-----
CREATE TABLE IF NOT EXISTS ITEM_DETAILS
(
ITEMID INTEGER,
ORDID INTEGER,
ITEMNAME CHARACTER VARYING(200)
);

CREATE TABLE IF NOT EXISTS ORDER_DETAILS
(
ORDID INTEGER,
ORDNAME CHARACTER VARYING(200),
ORDEREDPLACE CHARACTER VARYING(55)
);

```

```
CREATE TABLE IF NOT EXISTS PAYMENT_DETAILS
(
    PAYID INTEGER,
    ORDID INTEGER,
    PAYPLACE CHARACTER VARYING(55)
);
```

Caso de uso 1 — Agrupamentos

Em um banco de dados, um agrupamento é um conjunto de regras para determinar como os dados são classificados e comparados. Um agrupamento geralmente é aplicado à forma como os dados de texto são classificados em diferentes idiomas para indexação e para fazer comparações entre valores de texto. Idiomas diferentes têm conjuntos de caracteres e ordenações diferentes. Com um agrupamento, você pode classificar os dados de caracteres de um determinado idioma usando regras que definem a sequência correta de caracteres. Também é possível especificar o seguinte:

- Sensibilidade a maiúsculas
- Marcas de acento
- Tipos de personagens Kana
- Uso de símbolos ou pontuação
- Largura de caracteres
- Classificação de palavras

Pode haver um impacto no desempenho se a coluna de junção usar um agrupamento diferente. O exemplo de consulta a seguir usa três tabelas, com um agrupamento diferente para a coluna de junção.

Nome da tabela	Nome da coluna
<code>rn_r_segment</code>	<code>pnr_number character varying(15) COLLATE pg_catalog."C" NOT NULL</code>
<code>rn_r_segment_pax</code>	<code>pnr_number character varying(15) COLLATE pg_catalog."default" NOT NULL</code>

`rnr_seat_numbers``pnr_number character varying(15)
COLLATE pg_catalog."default" NOT
NULL`

```
EXPLAIN ANALYZE SELECT  
A.PNR_NUMBER,  
A.PAX_ID,  
A.SEGMENT_ID,  
B.OANDD_ID,  
C.SEAT_ID,  
C.BD_AIRPORT_CODE,  
C.OFF_AIRPORT_CODE,  
C.SEAT_NUMBER ,  
B.CABIN_CLASS ,  
A.SEGMENT_PAX_ID,  
C.SEAT_ALLOC_ID,  
C.SSR_ID,  
C.SEAT_ATTRIBUTE_CODE  
from  
RNR_SEGMENT_PAX A,  
RNR_SEGMENT B,  
RNR_SEAT_NUMBERS C  
where  
B.AIRLINE_IATA_CODE = 'XX'  
and B.FLIGHT_CARRIER = 'XX'  
and B.FLIGHT_NUMBER = 140  
and B.FLIGHT_SUFFIX = '*'  
and B.FLIGHT_DATE_LTC = TO_DATE('01-JAN-2023', 'DD-MON-YYYY')  
and A.AIRLINE_IATA_CODE = B.AIRLINE_IATA_CODE  
and A.PNR_NUMBER = B.PNR_NUMBER  
and A.SEGMENT_ID = B.SEGMENT_ID  
and C.AIRLINE_IATA_CODE = B.AIRLINE_IATA_CODE  
and C.PNR_NUMBER = B.PNR_NUMBER  
and C.SEGMENT_ID = B.SEGMENT_ID  
and A.PAX_ID = C.PAX_ID  
and B.PNR_NUMBER in ('9F1588', 'E37DE0', '04E82B', '813D11', 'BFF10F');
```

O plano de consulta da consulta anterior usa uma varredura de sequência na `rnr_seat_numbers` tabela, mesmo que essa tabela tenha um índice adequado nas colunas unidas. O planejador não

está usando uma varredura de índice porque essas colunas unidas estão usando agrupamentos diferentes:

```
Nested Loop (cost=1112.14..927363.51 rows=1 width=833) (actual time=5395.367..5397.253
rows=0 loops=1)
  Join Filter: (((b.pnr_number)::text = (a.pnr_number)::text) AND (b.segment_id =
a.segment_id))
    -> Gather (cost=1111.58..670766.48 rows=1 width=843) (actual
time=5395.367..5397.251 rows=0 loops=1)
      Workers Planned: 2
      Workers Launched: 2
      -> Hash Join (cost=111.58..669766.38 rows=1 width=843) (actual
time=5388.992..5388.993 rows=0 loops=3)
        Hash Cond: (((c.pnr_number)::text = (b.pnr_number)::text) AND
(c.segment_id = b.segment_id))
          -> Parallel Seq Scan on rnr_seat_numbers c (cost=0.00..582154.96
rows=16666637 width=760) (actual time=0.008..2963.019 rows=13333333 loops=3)
            Filter: ((airline_iata_code)::text = 'XX'::text)
          -> Hash (cost=111.52..111.52 rows=4 width=86) (actual time=0.121..0.121
rows=2 loops=3)
            Buckets: 1024 Batches: 1 Memory Usage: 9kB
            -> Index Scan using rnr_segment_pk on rnr_segment b
(cost=0.56..111.52 rows=4 width=86) (actual time=0.082..0.116 rows=2 loops=3)
              Index Cond: (((pnr_number)::text = ANY
('{9F1588,E37DE0,04E82B,813D11,BFF10F}'::text[])) AND ((airline_iata_code)::text =
'XX'::text))
                Filter: (((flight_carrier)::text = 'XX'::text) AND
(flight_number = 140) AND ((flight_suffix)::text = '*'::text) AND (flight_date_ltc =
to_date('01-JAN-2023'::text, 'DD-MON-YYYY'::text)))
                Rows Removed by Filter: 20
            -> Index Scan using rnr_segment_pax_pk on rnr_segment_pax a (cost=0.56..256597.02
rows=1 width=28) (never executed)
              Index Cond: (((airline_iata_code)::text = 'XX'::text) AND (segment_id =
c.segment_id) AND (pax_id = c.pax_id))
              Filter: ((c.pnr_number)::text = (pnr_number)::text)
Planning Time: 0.982 ms
Execution Time: 5397.314 ms
```

Para alterar o agrupamento de colunas da tabela do "C" idioma para o agrupamento padrão fornecido pelo PostgreSQL, execute a instrução a seguir alter e, em seguida, analise a tabela:

```
alter table rnr_segment alter column pnr_number type character varying(15) COLLATE
pg_catalog."default";
```

Analyze rnr_segment;

O plano de consulta agora usa uma varredura de índice e o tempo de execução é reduzido.

```
Nested Loop (cost=1.69..146.63 rows=1 width=833) (actual time=0.155..0.155 rows=0
 loops=1)
  -> Nested Loop (cost=1.13..145.89 rows=1 width=111) (actual time=0.154..0.155
 rows=0 loops=1)
    -> Index Scan using rnr_segment_pk on rnr_segment b (cost=0.56..111.51 rows=4
 width=86) (actual time=0.048..0.097 rows=2 loops=1)
      Index Cond: (((pnr_number)::text = ANY
 ('{9F1588,E37DE0,04E82B,813D11,BFF10F}'::text[])) AND ((airline_iata_code)::text =
 'XX'::text))
      Filter: (((flight_carrier)::text = 'XX'::text) AND (flight_number =
 140) AND ((flight_suffix)::text = '*'::text) AND (flight_date_ltc = to_date('01-
JAN-2023'::text, 'DD-MON-YYYY'::text)))
      Rows Removed by Filter: 20
    -> Index Scan using rnr_segment_pax_pk on rnr_segment_pax a (cost=0.56..8.58
 rows=1 width=28) (actual time=0.027..0.027 rows=0 loops=2)
      Index Cond: (((airline_iata_code)::text = 'XX'::text) AND
 ((pnr_number)::text = (b.pnr_number)::text) AND (segment_id = b.segment_id))
  -> Index Scan using rnr_seat_numbers_pk on rnr_seat_numbers c (cost=0.56..0.72
 rows=1 width=760) (never executed)
    Index Cond: (((pnr_number)::text = (a.pnr_number)::text) AND (segment_id =
 a.segment_id) AND (pax_id = a.pax_id) AND ((airline_iata_code)::text = 'XX'::text))
Planning Time: 1.432 ms
Execution Time: 0.207 ms
```

Caso de uso 2 — Incompatibilidade de tipos de dados

Escolher o tipo de dados adequado com base nos dados ajuda a fornecer o equilíbrio ideal entre tamanho e desempenho do armazenamento.

O exemplo de consulta a seguir usa a `pnr_number` coluna para unir duas tabelas. A `pnr_number` coluna tem tipos de dados diferentes em tabelas diferentes.

Nome da tabela

Nome da coluna e tipo de dados

`perf_user.rnr_segment_pax`

`pnr_number character varying(6)`

perf_user.rnr_expiry_date

pnr_number character(2)

```
EXPLAIN ANALYZE UPDATE perf_user.RNR_SEGMENT_PAX x SET ARC_EXPIRY_DATE =
y.ARC_EXPIRY_DATE
FROM (SELECT AIRLINE_IATA_CODE, PNR_NUMBER, ARC_EXPIRY_DATE, 0+row_num ROW_NUM
FROM perf_user.RNR_EXPIRY_DATE
WHERE airline_iata_code = 'XX'
AND row_num BETWEEN (1*5000)+0 AND (1+1)*5000) y
WHERE x.airline_iata_code = y.airline_iata_code
AND x.PNR_NUMBER =y.PNR_NUMBER;
```

```
Update on rnr_segment_pax x (cost=290.97..1104986.32 rows=15515 width=460) (actual
time=14574.118..14574.120 rows=0 loops=1)
-> Hash Join (cost=290.97..1104986.32 rows=15515 width=460) (actual
time=16.967..14101.983 rows=11953 loops=1)
Hash Cond: ((x.pnr_number)::text = (rnr_expiry_date.pnr_number)::text)
-> Seq Scan on rnr_segment_pax x (cost=0.00..954539.00 rows=40000320
width=446) (actual time=0.011..9702.989 rows=40000000 loops=1)
Filter: ((airline_iata_code)::bpchar = 'XX'::bpchar)
-> Hash (cost=225.37..225.37 rows=5248 width=24) (actual time=16.540..16.541
rows=5001 loops=1)
Buckets: 8192 Batches: 1 Memory Usage: 338kB
-> Index Scan using rnr_expiry_date_idx1 on rnr_expiry_date
(cost=0.29..225.37 rows=5248 width=24) (actual time=3.102..15.331 rows=5001 loops=1)
Index Cond: ((row_num >= 5000) AND (row_num <= 10000))
Filter: (airline_iata_code = 'XX'::bpchar)
Planning Time: 4.445 ms
Execution Time: 14574.322 ms
```

Quando você executa `EXPLAIN ANALYZE`, o planejador usa uma varredura de sequência ativada em `rnr_segment_pax` vez de uma varredura de índice, mesmo que as colunas usadas na junção tenham índices. O planejador não está usando uma varredura de índice porque as colunas usadas na junção têm comprimentos diferentes.

Altere as colunas da tabela para manter o mesmo tipo de dados para as duas tabelas envolvidas na condição de junção e, em seguida, analise a tabela:

```
alter table perf_user.rnr_expiry_date alter column airline_iata_code type character
varying(6) ;
```

```
analyze perf_user.rnr_expiry_date;
```

Agora, as tabelas têm o mesmo comprimento nas duas colunas usadas na condição de junção.

Execute EXPLAIN ANALYZE novamente. O planejador executa uma varredura de índice, o que melhora significativamente o desempenho da consulta.

```
Update on rnr_segment_pax x (cost=0.86..59733.09 rows=14637 width=460) (actual
time=416.653..416.654 rows=0 loops=1)
  -> Nested Loop (cost=0.86..59733.09 rows=14637 width=460) (actual
time=0.103..91.106 rows=11953 loops=1)
    -> Index Scan using rnr_expiry_date_idx1 on rnr_expiry_date
(cost=0.29..212.69 rows=4951 width=24) (actual time=0.025..3.023 rows=5001 loops=1)
      Index Cond: ((row_num >= 5000) AND (row_num <= 10000))
      Filter: ((airline_iata_code)::text = 'XX'::text)
    -> Index Scan using rnr_segment_pax_pk on rnr_segment_pax x (cost=0.56..11.99
rows=3 width=446) (actual time=0.014..0.016 rows=2 loops=5001)
      Index Cond: (((airline_iata_code)::text = 'XX'::text) AND
((pnr_number)::text = (rnr_expiry_date.pnr_number)::text))
Planning Time: 0.310 ms
Execution Time: 416.696 ms
```

Caso de uso 3 — Chamada de função na instrução SELECT

Chamar uma função em uma where cláusula pode reduzir o desempenho da consulta quando a função está, VOLATILE e você não usa a select palavra-chave ao chamar a função:

```
Select * from tab_name where FieldName = FunctionName(parameters);
```

Uma varredura de índice é executada se a select instrução for usada ao chamar a função:

```
Select * from tab_name where FieldName = ( select FunctionName(parameters) );
```

O pnr_number campo tem um índice na rnr_expiry_date tabela. O índice é usado ao comparar o valor na where cláusula.

```
explain analyze select * from perf_user.rnr_expiry_date where pnr_number= 'EE9F41';

"Index Scan using rnr_expiry_date_idx3 on rnr_expiry_date (cost=0.29..8.31 rows=1
width=72) (actual time=0.020..0.021 rows=1 loops=1)"
```

```
"  Index Cond: ((pnr_number)::text = 'EE9F41'::text)"
"Planning Time: 0.063 ms"
"Execution Time: 0.038 ms"
```

Uma varredura sequencial é executada quando uma função é chamada sem a `select` palavra-chave, mesmo quando um índice está disponível no campo.

```
explain analyze select * from perf_user.rnr_expiry_date where pnr_number=
perf_user.return_data();

"Seq Scan on rnr_expiry_date (cost=0.00..27084.00 rows=1 width=72) (actual
time=0.112..135.917 rows=1 loops=1)"
"  Filter: ((pnr_number)::text = (perf_user.return_data())::text)"
"  Rows Removed by Filter: 99999"
"Planning Time: 0.053 ms"
"Execution Time: 136.803 ms"
```

Uma varredura de índice é realizada quando a função é chamada com a `select` palavra-chave.

```
explain analyze select * from perf_user.rnr_expiry_date where pnr_number= (select
perf_user.return_data() );

"Index Scan using rnr_expiry_date_idx3 on rnr_expiry_date (cost=0.55..8.57 rows=1
width=72) (actual time=0.058..0.061 rows=1 loops=1)"
"  Index Cond: ((pnr_number)::text = ($0)::text)"
"  InitPlan 1 (returns $0)"
"    -> Result (cost=0.00..0.26 rows=1 width=32) (actual time=0.021..0.022 rows=1
loops=1)"
"Planning Time: 0.147 ms"
"Execution Time: 0.111 ms"
```

Caso de uso 4 — IN ou EXISTS

Se a consulta tiver `NOT IN` operadores `IN` ou, recomendamos verificar o plano de consulta para confirmar se o índice adequado está sendo usado. Se o índice adequado não estiver sendo usado e o desempenho da consulta estiver demorando mais do que o esperado, tente reescrever a consulta usando as `NOT EXISTS` condições `EXISTS` ou.

Considere o exemplo a seguir, que usa `NOT IN`:

```
EXPLAIN ANALYZE SELECT
```

```

TEST_VEH.TEST_VEH_ID,
TEST_VEH.VEHICLE_ID,
TEST_VEH.SERVICEPROGRAM_ID,
TEST_VEH.STARTDATE,
TEST_VEH.ENDDATE,
TEST_VEH.OILTYPE_ID
FROM PERF_USER.TEST_VEH TEST_VEH
JOIN PERF_USER.OILTYPE OT ON OT.OILTYPE_ID =TEST_VEH.OILTYPE_ID
JOIN PERF_USER.SERVICEPROGRAM SP ON SP.SERVICEPROGRAM_ID = TEST_VEH.SERVICEPROGRAM_ID
WHERE SP.PROGNAME = '18FCE8FDAF365BB'
      AND OT.OILTYPE_ID =3
      AND TEST_VEH.ENDDATE IS NOT NULL
      AND TEST_VEH.TEST_VEH_ID NOT IN
          (SELECT TEST_VEH_ID
           FROM PERF_USER.VEHICLESERVICEHISTORY
           WHERE TEST_VEH_ID > 1
          );

```

```

-----
"Nested Loop (cost=1009.16..1188860356305.01 rows=1 width=76) (actual
time=37299.891..37347.853 rows=0 loops=1)"
"  -> Gather (cost=1009.16..1188860356303.88 rows=1 width=76) (actual
time=37299.890..37347.849 rows=0 loops=1)"
"      Workers Planned: 2"
"      Workers Launched: 2"
"      -> Hash Join (cost=9.16..1188860355303.78 rows=1 width=76) (actual
time=37286.742..37286.751 rows=0 loops=3)"
"          Hash Cond: ((test_veh.serviceprogram_id)::text =
(sp.serviceprogram_id)::text)"
"              -> Parallel Index Scan using test_veh_oiltype_id_ind on test_veh
(cost=0.56..1188860351273.04 rows=1072570 width=76) (actual time=37276.290..37276.292
rows=1 loops=3)"
"                  Index Cond: (oiltype_id = 3)"
"                  Filter: ((enddate IS NOT NULL) AND (NOT (SubPlan 1)))"
"                  Rows Removed by Filter: 0"
"                  SubPlan 1"
"                      -> Materialize (cost=0.00..1025071.31 rows=3333332 width=8)
(actual time=0.418..23201.432 rows=25001498 loops=4)"
"                          -> Seq Scan on vehicleservicehistory
(cost=0.00..728195.65 rows=3333332 width=8) (actual time=0.416..13249.975
rows=25001498 loops=4)"
"                              Filter: (test_veh_id > 1)"
"                                  -> Hash (cost=8.58..8.58 rows=1 width=11) (actual time=9.045..9.046
rows=0 loops=3)"
"                                      Buckets: 1024 Batches: 1 Memory Usage: 8kB"

```

```

"          -> Index Scan using progname_id_ind on serviceprogram sp
(cost=0.56..8.58 rows=1 width=11) (actual time=9.043..9.044 rows=0 loops=3)"
"          Index Cond: ((progname)::text = '18FCE8FDAF365BB'::text)"
" -> Seq Scan on oiltype ot (cost=0.00..1.12 rows=1 width=8) (never executed)"
"          Filter: (oiltype_id = 3)"
"Planning Time: 37.696 ms"
"Execution Time: 37366.335 ms"

```

A consulta está levando mais de 37 segundos e 366 milissegundos para recuperar 4 milhões de registros.

O plano de consulta indica que uma varredura de sequência é executada na tabela usada na subconsultavehicleservicehistory. A varredura da sequência está produzindo um grande número de registros. Para cada um desses registros na subconsulta, a consulta está executando uma varredura completa da tabela, o que está causando o problema de desempenho.

Para evitar a varredura de sequência na subconsulta, reescreva a subconsulta para usar uma subconsulta correlacionada com. NOT EXISTS A subconsulta correlacionada usará uma varredura de índice e um número reduzido de varreduras de tabela:

```

EXPLAIN ANALYZE SELECT
    TEST_VEH.TEST_VEH_ID,
    TEST_VEH.VEHICLE_ID,
    TEST_VEH.SERVICEPROGRAM_ID,
    TEST_VEH.STARTDATE,
    TEST_VEH.ENDDATE,
    TEST_VEH.OILTYPE_ID
FROM PERF_USER.TEST_VEH TEST_VEH
JOIN PERF_USER.OILTYPE OT ON OT.OILTYPE_ID =TEST_VEH.OILTYPE_ID
JOIN PERF_USER.SERVICEPROGRAM SP ON SP.SERVICEPROGRAM_ID = TEST_VEH.SERVICEPROGRAM_ID
WHERE SP.PROGNAME = '18FCE8FDAF365BB'
    AND OT.OILTYPE_ID =3
    AND TEST_VEH.ENDDATE IS NOT NULL
    AND NOT EXISTS
        (SELECT TEST_VEH_ID
         FROM PERF_USER.VEHICLESERVICEHISTORY
         WHERE
TEST_VEH.TEST_VEH_ID=VEHICLESERVICEHISTORY.TEST_VEH_ID
         AND TEST_VEH_ID > 1
        );
-----

```

```

"Nested Loop Anti Join (cost=1009.03..936146.10 rows=1 width=76) (actual
time=12.693..12.810 rows=0 loops=1)"
"  -> Nested Loop (cost=1008.59..936141.78 rows=1 width=76) (actual
time=12.692..12.809 rows=0 loops=1)"
"    -> Gather (cost=1008.59..936140.64 rows=1 width=76) (actual
time=12.691..12.807 rows=0 loops=1)"
"      Workers Planned: 2"
"      Workers Launched: 2"
"    -> Hash Join (cost=8.59..935140.54 rows=1 width=76) (actual
time=0.773..0.774 rows=0 loops=3)"
"      Hash Cond: ((test_veh.serviceprogram_id)::text =
(sp.serviceprogram_id)::text)"
"        -> Parallel Seq Scan on test_veh (cost=0.00..927087.67
rows=2145139 width=76) (actual time=0.672..0.672 rows=1 loops=3)"
"          Filter: ((enddate IS NOT NULL) AND (oiltype_id = 3))"
"          Rows Removed by Filter: 7"
"        -> Hash (cost=8.58..8.58 rows=1 width=11) (actual
time=0.040..0.040 rows=0 loops=3)"
"          Buckets: 1024 Batches: 1 Memory Usage: 8kB"
"        -> Index Scan using proname_id_ind on serviceprogram sp
(cost=0.56..8.58 rows=1 width=11) (actual time=0.039..0.040 rows=0 loops=3)"
"          Index Cond: ((proname)::text =
'18FCE8FDAF365BB'::text)"
"        -> Seq Scan on oiltype ot (cost=0.00..1.12 rows=1 width=8) (never executed)"
"          Filter: (oiltype_id = 3)"
"    -> Index Only Scan using veh_ser_ind on vehicleservicehistory (cost=0.44..4.32
rows=1 width=8) (never executed)"
"      Index Cond: ((test_veh_id = test_veh.test_veh_id) AND (test_veh_id > 1))"
"      Heap Fetches: 0"
"Planning Time: 11.115 ms"
"Execution Time: 12.871 ms"

```

Após a modificação, a consulta está levando menos de 13 ms para processar 4 milhões de registros

De acordo com o plano de consulta da consulta modificada, a tabela `vehicleservicehistory` pode ter uma varredura de índice. Usar uma varredura de índice reduz o custo e o número de linhas afetadas. Dessa forma, você pode reduzir o tempo de execução de uma consulta e aumentar seu desempenho.

Caso de uso 5 — Subconsultas ou CTEs

Expressões de tabela comuns (CTEs) ajudam a dividir consultas grandes em consultas menores. Isso facilita a manutenção de toda a consulta.

As junções de subconsultas são substituídas por junções CTE, que são mais legíveis porque a consulta é nomeada e separada dentro da seção CTE. Isso é especialmente útil quando o tamanho da consulta aumenta e a manutenção da consulta se torna mais difícil. Além disso, os resultados do CTE no PostgreSQL são materializados. Se você chamar o CTE em vários lugares, a definição real da consulta será executada somente uma vez. O resultado será armazenado na memória. Você pode usar isso para qualquer lógica complexa que deva ser usada em vários lugares na mesma consulta. Coloque essa lógica dentro de um CTE e chame o CTE quantas vezes quiser.

Por exemplo, um cliente estava usando consultas de aplicativos em linha com muitas subconsultas dentro das consultas. As subconsultas foram filtradas pelos valores dos parâmetros de entrada enviados pelos aplicativos.

```
EXPLAIN ANALYZE
SELECT * FROM
ORDER_DETAILS A
WHERE A.ORDID IN (SELECT ORDID FROM PAYMENT_DETAILS)
AND A.ORDID IN (SELECT ORDID FROM ITEM_DETAILS )
AND A.ORDID = 1000000;
```

```
"Nested Loop Semi Join (cost=3000.00..194258.21 rows=5 width=74) (actual
time=201.605..747.945 rows=5 loops=1)"
"  -> Nested Loop Semi Join (cost=2000.00..135040.47 rows=5 width=74) (actual
time=146.016..666.779 rows=5 loops=1)"
"      -> Gather (cost=1000.00..78580.31 rows=5 width=74) (actual
time=58.893..463.570 rows=5 loops=1)"
"          Workers Planned: 2"
"          Workers Launched: 2"
"      -> Parallel Seq Scan on order_details a (cost=0.00..77579.81 rows=2
width=74) (actual time=165.627..549.702 rows=2 loops=3)"
"          Filter: (ordid = 1000000)"
"          Rows Removed by Filter: 1666665"
"      -> Materialize (cost=1000.00..56460.07 rows=3 width=4) (actual
time=17.424..40.638 rows=1 loops=5)"
"          -> Gather (cost=1000.00..56460.06 rows=3 width=4) (actual
time=87.113..203.178 rows=1 loops=1)"
"          Workers Planned: 2"
```

```

"           Workers Launched: 2"
"           -> Parallel Seq Scan on payment_details (cost=0.00..55459.76
rows=1 width=4) (actual time=174.431..423.792 rows=1 loops=3)"
"           Filter: (ordid = 1000000)"
"           Rows Removed by Filter: 1333002"
"   -> Materialize (cost=1000.00..59217.64 rows=4 width=4) (actual time=11.117..16.231
rows=1 loops=5)"
"       -> Gather (cost=1000.00..59217.62 rows=4 width=4) (actual
time=55.581..81.148 rows=1 loops=1)"
"           Workers Planned: 2"
"           Workers Launched: 2"
"           -> Parallel Seq Scan on item_details (cost=0.00..58217.22 rows=2
width=4) (actual time=287.030..411.004 rows=1 loops=3)"
"           Filter: (ordid = 1000000)"
"           Rows Removed by Filter: 1333080"
"Planning Time: 0.266 ms"
"Execution Time: 747.986 ms"

```

Depois de modificar as subconsultas usando um CTE e adicionando filtros para que somente os conjuntos de linhas necessários sejam recuperados, o desempenho da consulta melhora.

```

EXPLAIN ANALYZE
WITH PAYMENT AS
(
  SELECT * FROM PAYMENT_DETAILS WHERE  ORDDID = 1000000
),
ITEM AS
(SELECT * FROM ITEM_DETAILS  WHERE  ORDDID = 1000000)
SELECT * FROM
ORDER_DETAILS A JOIN PAYMENT B
ON A.ORDID=B.ORDID
JOIN ITEM C ON B.ORDID=C.ORDID

```

```

"Nested Loop (cost=3000.00..194258.91 rows=60 width=166) (actual time=586.410..732.918
rows=80 loops=1)"
"   -> Nested Loop (cost=2000.00..115677.83 rows=12 width=92) (actual
time=456.760..457.083 rows=16 loops=1)"
"       -> Gather (cost=1000.00..59217.62 rows=4 width=48) (actual
time=153.802..154.060 rows=4 loops=1)"
"           Workers Planned: 2"
"           Workers Launched: 2"

```

```

"          -> Parallel Seq Scan on item_details (cost=0.00..58217.22 rows=2
width=48) (actual time=85.417..249.045 rows=1 loops=3)"
"              Filter: (ordid = 1000000)"
"              Rows Removed by Filter: 1333332"
"          -> Materialize (cost=1000.00..56460.07 rows=3 width=44) (actual
time=75.738..75.753 rows=4 loops=4)"
"          -> Gather (cost=1000.00..56460.06 rows=3 width=44) (actual
time=302.947..303.005 rows=4 loops=1)"
"              Workers Planned: 2"
"              Workers Launched: 2"
"          -> Parallel Seq Scan on payment_details (cost=0.00..55459.76
rows=1 width=44) (actual time=184.609..294.784 rows=1 loops=3)"
"              Filter: (ordid = 1000000)"
"              Rows Removed by Filter: 1333332"
"  -> Materialize (cost=1000.00..78580.34 rows=5 width=74) (actual time=8.103..17.238
rows=5 loops=16)"
"          -> Gather (cost=1000.00..78580.31 rows=5 width=74) (actual
time=129.641..275.795 rows=5 loops=1)"
"              Workers Planned: 2"
"              Workers Launched: 2"
"          -> Parallel Seq Scan on order_details a (cost=0.00..77579.81 rows=2
width=74) (actual time=78.556..268.994 rows=2 loops=3)"
"              Filter: (ordid = 1000000)"
"              Rows Removed by Filter: 1666665"
"Planning Time: 0.108 ms"
"Execution Time: 732.953 ms"

```

Essas são as observações dos dados de exemplo. Quando você executa a consulta em um grande conjunto de dados, a diferença no desempenho será muito alta.

Perguntas frequentes

Encontre respostas para perguntas frequentes sobre o ajuste do desempenho da consulta.

O que é EXPLAIN?

EXPLAIN é uma palavra-chave que você acrescenta a uma consulta PostgreSQL (SELECT,, UPDATE, INSERT, DELETE) para gerar um plano de consulta. O plano de consulta do PostgreSQL detalha como o banco de dados pretende executar a consulta. Esse plano inclui informações sobre a ordem de uma varredura de tabela, uso de índices e junções.

Use o plano de consulta para identificar possíveis gargalos, otimizar consultas e melhorar o desempenho geral. Ao analisar o plano de consulta, considere os seguintes fatores:

- Abordagens de acesso à tabela
- Junte abordagens
- Condições do filtro
- Operações de classificação
- Uso do índice
- Paralelismo
- Statistics
- Estimativas de custo
- Linhas recuperadas de cada etapa
- Distribuição de dados

Para obter mais informações sobre [EXPLAIN](#), consulte a documentação do PostgreSQL.

O que é EXPLAIN ANALYZE?

Quando você adiciona um prefixo EXPLAIN ANALYZE a uma consulta e executa a consulta, o PostgreSQL executa a consulta e retorna o plano de consulta e as estatísticas de tempo de execução. O tempo de execução real, as linhas processadas em cada etapa e outras informações relevantes são exibidas junto com o plano de consulta. O uso EXPLAIN ANALYZE em um banco

de dados de produção deve ser feito com cuidado, pois a execução da consulta pode afetar o desempenho do banco de dados durante a análise.

Para obter mais informações sobre [EXPLAIN ANALYZE](#), consulte a documentação do PostgreSQL.

O que é agrupamento no PostgreSQL?

No PostgreSQL, um agrupamento é um conjunto de regras para determinar como as strings são comparadas e classificadas. O agrupamento define a ordem na qual os caracteres são considerados nas comparações, considerando as regras e conversões específicas do idioma.

Para obter mais informações sobre [agrupamento](#), consulte a documentação do PostgreSQL.

O que é um CTE?

Em um banco de dados PostgreSQL, uma Expressão de Tabela Comum (CTE) é um conjunto de resultados temporário nomeado que você pode referenciar. Os CTEs fornecem uma maneira de criar consultas SQL mais legíveis e modulares dividindo a lógica complexa em unidades menores e nomeadas.

Para obter mais informações sobre [CTEs](#), consulte a documentação do PostgreSQL.

Quais são as categorias de funções no PostgreSQL?

Cada função do PostgreSQL tem uma classificação de volatilidade, com as possibilidades VOLATILE sendo,, ou: STABLE IMMUTABLE

- **VOLÁTIL** — Uma VOLATILE função pode fazer qualquer coisa, inclusive modificar o banco de dados. Ele pode retornar resultados diferentes em chamadas sucessivas com os mesmos argumentos. O otimizador não faz suposições sobre o comportamento de tais funções. Uma consulta usando uma função volátil reavaliará a função em cada linha em que seu valor for necessário.
- **ESTÁVEL** — Uma STABLE função não pode modificar o banco de dados. É garantido que retornará os mesmos resultados com os mesmos argumentos para todas as linhas em uma única instrução. Quando você usa essa classificação, o otimizador pode otimizar várias chamadas da função para uma única chamada. Em particular, é seguro usar uma expressão que contenha essa função em uma condição de varredura de índice. (Como uma varredura de índice avaliará o valor

de comparação apenas uma vez, não uma vez em cada linha, não é válido usar uma VOLATILE função em uma condição de varredura de índice.)

- **IMUTÁVEL** — Uma IMMUTABLE função não pode modificar o banco de dados e tem a garantia de retornar os mesmos resultados com os mesmos argumentos para sempre. Quando você usa essa classificação, o otimizador pode pré-avaliar a função quando uma consulta a chama com argumentos constantes. Por exemplo, uma consulta como a `SELECT ... WHERE x = 2 + 2` pode ser simplificada à primeira vista `SELECT ... WHERE x = 4`, porque a função subjacente ao operador de adição de números inteiros está marcada IMMUTABLE.

VOLATILE é o padrão se o `CREATE FUNCTION` comando não especificar uma categoria. Para obter mais informações sobre os [tipos de função](#), consulte a documentação do PostgreSQL.

Recursos

Referências

- [EXPLIQUE](#)
- [Usando EXPLAIN](#)
- [Support de agrupamento](#)
- [Consultas WITH \(expressões de tabela comuns\)](#)

Guias

- [Atividades de manutenção de bancos de dados PostgreSQL no Amazon RDS e no Amazon Aurora para evitar problemas de desempenho](#)
- [Ajuste dos parâmetros do PostgreSQL no Amazon RDS e no Amazon Aurora](#)

Colaboradores

Os colaboradores deste documento incluem:

- Tirumala Dasari, consultora líder — bancos de dados, AWS
- Veeranjanyulu Grandhi, consultor líder — bancos de dados, AWS
- Vamsikrishna Jammula, consultor — bancos de dados, AWS
- Srinivas Potlachervoo, consultor sênior líder — bancos de dados, AWS
- Naga Srinivas Reddy Ravulapati, consultora — bancos de dados, AWS

Histórico do documento

A tabela a seguir descreve alterações significativas feitas neste guia. Se desejar receber notificações sobre futuras atualizações, inscreva-se em um [feed RSS](#).

Alteração	Descrição	Data
Publicação inicial	—	23 de abril de 2024

AWS Glossário de orientação prescritiva

A seguir estão os termos comumente usados em estratégias, guias e padrões fornecidos pela Orientação AWS Prescritiva. Para sugerir entradas, use o link Fornecer feedback no final do glossário.

Números

7 Rs

Sete estratégias comuns de migração para mover aplicações para a nuvem. Essas estratégias baseiam-se nos 5 Rs identificados pela Gartner em 2011 e consistem em:

- Refatorar/rearquitetar: mova uma aplicação e modifique sua arquitetura aproveitando ao máximo os recursos nativos de nuvem para melhorar a agilidade, a performance e a escalabilidade. Isso normalmente envolve a portabilidade do sistema operacional e do banco de dados. Exemplo: migrar seu banco de dados Oracle on-premises para o Amazon Aurora Edição Compatível com PostgreSQL.
- Redefinir a plataforma (mover e redefinir [mover e redefinir (lift-and-reshape)]): mova uma aplicação para a nuvem e introduza algum nível de otimização a fim de aproveitar os recursos da nuvem. Exemplo: migrar seu banco de dados Oracle on-premises para o Amazon Relational Database Service (Amazon RDS) para Oracle na Nuvem AWS.
- Recomprar (drop and shop): mude para um produto diferente, normalmente migrando de uma licença tradicional para um modelo SaaS. Exemplo: migrar seu sistema de gerenciamento de relacionamento com o cliente (CRM) para o Salesforce.com.
- Redefinir a hospedagem (mover sem alterações [lift-and-shift]): mover uma aplicação para a nuvem sem fazer nenhuma alteração a fim de aproveitar os recursos da nuvem. Exemplo: migrar seu banco de dados Oracle on-premises para o Oracle em uma instância do EC2 na Nuvem AWS.
- Realocar (mover o hipervisor sem alterações [hypervisor-level lift-and-shift]): mover a infraestrutura para a nuvem sem comprar novo hardware, reescrever aplicações ou modificar suas operações existentes. Você migra servidores de uma plataforma on-premises para um serviço de nuvem para a mesma plataforma. Exemplo: migrar um Microsoft Hyper-V aplicativo para o AWS.
- Reter (revisitar): mantenha as aplicações em seu ambiente de origem. Isso pode incluir aplicações que exigem grande refatoração, e você deseja adiar esse trabalho para um

momento posterior, e aplicações antigas que você deseja manter porque não há justificativa comercial para migrá-las.

- Retirar: desative ou remova aplicações que não são mais necessárias em seu ambiente de origem.

A

ABAC

Consulte [controle de acesso baseado em atributo](#).

serviços abstraídos

Veja [serviços gerenciados](#).

ACID

Veja [atomicidade, consistência, isolamento, durabilidade](#).

migração ativa-ativa

Um método de migração de banco de dados no qual os bancos de dados de origem e de destino são mantidos em sincronia (por meio de uma ferramenta de replicação bidirecional ou operações de gravação dupla), e ambos os bancos de dados lidam com transações de aplicações conectadas durante a migração. Esse método oferece suporte à migração em lotes pequenos e controlados, em vez de exigir uma substituição única. É mais flexível, mas exige mais trabalho do que a [migração ativa-passiva](#).

migração ativa-passiva

Um método de migração de banco de dados em que os bancos de dados de origem e de destino são mantidos em sincronia, mas somente o banco de dados de origem manipula as transações das aplicações conectadas, enquanto os dados são replicados no banco de dados de destino. O banco de dados de destino não aceita nenhuma transação durante a migração.

AGGREGATE FUNCTION

Uma função SQL que opera em um grupo de linhas e calcula um único valor de retorno para o grupo. Exemplos de funções agregadas incluem SUM e MAX.

AI

Veja [inteligência artificial](#).

AIOps

Veja [operações de inteligência artificial](#).

anonimização

O processo de excluir permanentemente informações pessoais em um conjunto de dados. A anonimização pode ajudar a proteger a privacidade pessoal. Dados anônimos não são mais considerados dados pessoais.

antipadrões

Uma solução frequentemente usada para um problema recorrente em que a solução é contraproducente, ineficaz ou menos eficaz do que uma alternativa.

controle de aplicações

Uma abordagem de segurança que permite o uso somente de aplicações aprovadas para ajudar a proteger um sistema contra malware.

portfólio de aplicações

Uma coleção de informações detalhadas sobre cada aplicação usada por uma organização, incluindo o custo para criar e manter a aplicação e seu valor comercial. Essas informações são fundamentais para [o processo de descoberta e análise de portfólio](#) e ajudam a identificar e priorizar as aplicações a serem migradas, modernizadas e otimizadas.

inteligência artificial (IA)

O campo da ciência da computação que se dedica ao uso de tecnologias de computação para desempenhar funções cognitivas normalmente associadas aos humanos, como aprender, resolver problemas e reconhecer padrões. Para obter mais informações, consulte [O que é inteligência artificial?](#)

operações de inteligência artificial (AIOps)

O processo de usar técnicas de machine learning para resolver problemas operacionais, reduzir incidentes operacionais e intervenção humana e aumentar a qualidade do serviço. Para obter mais informações sobre como AIOps é usado na estratégia de AWS migração, consulte o [guia de integração de operações](#).

criptografia assimétrica

Um algoritmo de criptografia que usa um par de chaves, uma chave pública para criptografia e uma chave privada para descriptografia. É possível compartilhar a chave pública porque ela não é usada na descriptografia, mas o acesso à chave privada deve ser altamente restrito.

atomicidade, consistência, isolamento, durabilidade (ACID)

Um conjunto de propriedades de software que garantem a validade dos dados e a confiabilidade operacional de um banco de dados, mesmo no caso de erros, falhas de energia ou outros problemas.

controle de acesso por atributo (ABAC)

A prática de criar permissões minuciosas com base nos atributos do usuário, como departamento, cargo e nome da equipe. Para obter mais informações, consulte [ABAC AWS](#) na documentação AWS Identity and Access Management (IAM).

fonte de dados autorizada

Um local onde você armazena a versão principal dos dados, que é considerada a fonte de informações mais confiável. Você pode copiar dados da fonte de dados autorizada para outros locais com o objetivo de processar ou modificar os dados, como anonimizá-los, redigi-los ou pseudonimizá-los.

Zona de disponibilidade

Um local distinto dentro de um Região da AWS que está isolado de falhas em outras zonas de disponibilidade e fornece conectividade de rede barata e de baixa latência a outras zonas de disponibilidade na mesma região.

AWS Estrutura de adoção da nuvem (AWS CAF)

Uma estrutura de diretrizes e melhores práticas AWS para ajudar as organizações a desenvolver um plano eficiente e eficaz para migrar com sucesso para a nuvem. AWS O CAF organiza a orientação em seis áreas de foco chamadas perspectivas: negócios, pessoas, governança, plataforma, segurança e operações. As perspectivas de negócios, pessoas e governança têm como foco habilidades e processos de negócios; as perspectivas de plataforma, segurança e operações concentram-se em habilidades e processos técnicos. Por exemplo, a perspectiva das pessoas tem como alvo as partes interessadas que lidam com recursos humanos (RH), funções de pessoal e gerenciamento de pessoal. Nessa perspectiva, o AWS CAF fornece orientação para desenvolvimento, treinamento e comunicação de pessoas para ajudar a preparar a organização para a adoção bem-sucedida da nuvem. Para obter mais informações, consulte o [site da AWS CAF](#) e o [whitepaper da AWS CAF](#).

AWS Estrutura de qualificação da carga de trabalho (AWS WQF)

Uma ferramenta que avalia as cargas de trabalho de migração do banco de dados, recomenda estratégias de migração e fornece estimativas de trabalho. AWS O WQF está incluído com AWS

Schema Conversion Tool (AWS SCT). Ela analisa esquemas de banco de dados e objetos de código, código de aplicações, dependências e características de performance, além de fornecer relatórios de avaliação.

B

bot malicioso

Um [bot](#) destinado a causar disrupção ou danos a indivíduos ou organizações.

BCP

Veja [planejamento de continuidade de negócios](#)

gráfico de comportamento

Uma visualização unificada e interativa do comportamento e das interações de recursos ao longo do tempo. É possível usar um gráfico de comportamento com o Amazon Detective para examinar tentativas de login malsucedidas, chamadas de API suspeitas e ações similares. Para obter mais informações, consulte [Dados em um gráfico de comportamento](#) na documentação do Detective.

sistema big-endian

Um sistema que armazena o byte mais significativo antes. Veja também [endianness](#).

classificação binária

Um processo que prevê um resultado binário (uma de duas classes possíveis). Por exemplo, seu modelo de ML pode precisar prever problemas como “Este e-mail é ou não é spam?” ou “Este produto é um livro ou um carro?”

filtro de bloom

Uma estrutura de dados probabilística e eficiente em termos de memória que é usada para testar se um elemento é membro de um conjunto.

blue/green deployment (implantação azul/verde)

Uma estratégia de implantação em que você cria dois ambientes separados, mas idênticos. Você executa a versão atual da aplicação em um ambiente (azul) e a nova versão da aplicação no outro ambiente (verde). Essa estratégia ajuda você a reverter rapidamente com o mínimo de impacto.

bot

Uma aplicação de software que executa tarefas automatizadas na internet e simula a atividade ou interação humana. Alguns bots são úteis ou benéficos, como crawlers da web que indexam informações na internet. Outros bots, conhecidos como bots maliciosos, têm como objetivo causar disrupção ou danos a indivíduos ou organizações.

botnet

Redes de [bots](#) infectadas por [malware](#) e sob o controle de uma única parte, conhecidas como bot herder ou operador de bots. Os botnets são o mecanismo mais conhecido para escalar bots e seu impacto.

ramo

Uma área contida de um repositório de código. A primeira ramificação criada em um repositório é a ramificação principal. Você pode criar uma nova ramificação a partir de uma ramificação existente e, em seguida, desenvolver recursos ou corrigir bugs na nova ramificação. Uma ramificação que você cria para gerar um recurso é comumente chamada de ramificação de recurso. Quando o recurso estiver pronto para lançamento, você mesclará a ramificação do recurso de volta com a ramificação principal. Para obter mais informações, consulte [Sobre filiais](#) (GitHub documentação).

Acesso de emergência

Em circunstâncias excepcionais e por meio de um processo aprovado, um meio rápido para um usuário obter acesso a um Conta da AWS que ele normalmente não tem permissão para acessar. Para obter mais informações, consulte o indicador [Implement break-glass procedures](#) nas orientações do AWS Well-Architected.

estratégia brownfield

A infraestrutura existente em seu ambiente. Ao adotar uma estratégia brownfield para uma arquitetura de sistema, você desenvolve a arquitetura de acordo com as restrições dos sistemas e da infraestrutura atuais. Se estiver expandindo a infraestrutura existente, poderá combinar as estratégias brownfield e [greenfield](#).

cache do buffer

A área da memória em que os dados acessados com mais frequência são armazenados.

capacidade de negócios

O que uma empresa faz para gerar valor (por exemplo, vendas, atendimento ao cliente ou marketing). As arquiteturas de microsserviços e as decisões de desenvolvimento podem

ser orientadas por recursos de negócios. Para obter mais informações, consulte a seção [Organizados de acordo com as capacidades de negócios](#) do whitepaper [Executar microsserviços containerizados na AWS](#).

planejamento de continuidade de negócios (BCP)

Um plano que aborda o impacto potencial de um evento disruptivo, como uma migração em grande escala, nas operações e permite que uma empresa retome as operações rapidamente.

C

CAF

Veja [AWS Cloud Adoption Framework](#).

implantação canário

O lançamento lento e incremental de uma versão para usuários finais. Quando estiver confiante, você implanta a nova versão e substitui a versão atual por completo.

CCoE

Veja [Centro de Excelência da Nuvem](#).

CDC

Veja [captura de dados de alteração](#).

captura de dados de alterações (CDC)

O processo de rastrear alterações em uma fonte de dados, como uma tabela de banco de dados, e registrar metadados sobre a alteração. É possível usar o CDC para várias finalidades, como auditar ou replicar alterações em um sistema de destino para manter a sincronização.

engenharia do caos

Introduzir intencionalmente falhas ou eventos disruptivos para testar a resiliência de um sistema. Você pode usar [AWS Fault Injection Service \(AWS FIS\)](#) para realizar experimentos que estressam suas AWS cargas de trabalho e avaliar sua resposta.

CI/CD

Veja [integração e entrega contínuas](#).

classificação

Um processo de categorização que ajuda a gerar previsões. Os modelos de ML para problemas de classificação predizem um valor discreto. Os valores discretos são sempre diferentes uns dos outros. Por exemplo, um modelo pode precisar avaliar se há ou não um carro em uma imagem.

criptografia no lado do cliente

Criptografia de dados localmente, antes que o alvo os AWS service (Serviço da AWS) receba.

Centro de excelência em nuvem (CCoE)

Uma equipe multidisciplinar que impulsiona os esforços de adoção da nuvem em toda a organização, incluindo o desenvolvimento de práticas recomendadas de nuvem, a mobilização de recursos, o estabelecimento de cronogramas de migração e a liderança da organização em transformações em grande escala. Para obter mais informações, consulte as [publicações CCo E](#) no blog de estratégia Nuvem AWS corporativa.

computação em nuvem

A tecnologia de nuvem normalmente usada para armazenamento de dados remoto e gerenciamento de dispositivos de IoT. A computação em nuvem é normalmente conectada à tecnologia de [computação de borda](#).

modelo operacional em nuvem

Em uma organização de TI, o modelo operacional usado para criar, amadurecer e otimizar um ou mais ambientes de nuvem. Para obter mais informações, consulte [Criar seu modelo operacional de nuvem](#).

estágios de adoção da nuvem

As quatro fases pelas quais as organizações normalmente passam ao migrar para a Nuvem AWS:

- Projeto: executar alguns projetos relacionados à nuvem para fins de prova de conceito e aprendizado
- Fundação — Fazer investimentos fundamentais para escalar sua adoção da nuvem (por exemplo, criar uma landing zone, definir um CCo E, estabelecer um modelo de operações)
- Migração: migrar aplicações individuais
- Reinvenção: otimizar produtos e serviços e inovar na nuvem

Esses estágios foram definidos por Stephen Orban na postagem do blog [The Journey Toward Cloud-First & the Stages of Adoption](#) no blog de estratégia Nuvem AWS empresarial. Para obter

informações sobre como eles se relacionam com a estratégia de AWS migração, consulte o [guia de preparação para migração](#).

CMDB

Veja [banco de dados de gerenciamento de configuração](#).

repositório de código

Um local onde o código-fonte e outros ativos, como documentação, amostras e scripts, são armazenados e atualizados por meio de processos de controle de versão. Os repositórios de nuvem comuns incluem o GitHub ou o Bitbucket Cloud. Cada versão do código é chamada de ramificação. Em uma estrutura de microsserviços, cada repositório é dedicado a uma única peça de funcionalidade. Um único pipeline de CI/CD pode usar vários repositórios.

cache frio

Um cache de buffer que está vazio, não está bem preenchido ou contém dados obsoletos ou irrelevantes. Isso afeta a performance porque a instância do banco de dados deve ler da memória principal ou do disco, um processo que é mais lento do que a leitura do cache do buffer.

dados frios

Dados que raramente são acessados e geralmente são históricos. Ao consultar esse tipo de dados, consultas lentas geralmente são aceitáveis. Mover esses dados para níveis ou classes de armazenamento de baixo desempenho e menos caros pode reduzir os custos.

visão computacional (CV)

Um campo de [IA](#) que usa machine learning para analisar e extrair informações de formatos visuais, como vídeos e imagens digitais. Por exemplo, a Amazon SageMaker AI fornece algoritmos de processamento de imagem para CV.

desvio de configuração

Em uma workload, uma alteração de configuração em relação ao estado esperado. Isso pode fazer com que a workload se torne incompatível e, normalmente, é gradual e não intencional.

banco de dados de gerenciamento de configuração (CMDB)

Um repositório que armazena e gerencia informações sobre um banco de dados e seu ambiente de TI, incluindo componentes de hardware e software e suas configurações. Normalmente, os dados de um CMDB são usados no estágio de descoberta e análise do portfólio da migração.

pacote de conformidade

Uma coleção de AWS Config regras e ações de remediação que você pode montar para personalizar suas verificações de conformidade e segurança. Você pode implantar um pacote de conformidade como uma entidade única em uma Conta da AWS região ou em uma organização usando um modelo YAML. Para obter mais informações, consulte [Pacotes de conformidade na documentação](#). AWS Config

integração contínua e entrega contínua (CI/CD)

O processo de automatizar os estágios de origem, criação, teste, preparação e produção do processo de lançamento do software. CI/CD é comumente descrito como um pipeline. CI/CD pode ajudá-lo a automatizar processos, melhorar a produtividade, melhorar a qualidade do código e entregar com mais rapidez. Para obter mais informações, consulte [Benefícios da entrega contínua](#). CD também pode significar implantação contínua. Para obter mais informações, consulte [Entrega contínua versus implantação contínua](#).

CV

Veja [visão computacional](#).

D

dados em repouso

Dados estacionários em sua rede, por exemplo, dados que estão em um armazenamento.

classificação de dados

Um processo para identificar e categorizar os dados em sua rede com base em criticalidade e confidencialidade. É um componente crítico de qualquer estratégia de gerenciamento de riscos de segurança cibernética, pois ajuda a determinar os controles adequados de proteção e retenção para os dados. A classificação de dados é um componente do pilar de segurança no AWS Well-Architected Framework. Para obter mais informações, consulte [Classificação de dados](#).

desvio de dados

Uma variação significativa entre os dados de produção e os dados usados para treinar um modelo de ML ou uma alteração significativa nos dados de entrada ao longo do tempo. O desvio de dados pode reduzir a qualidade geral, a precisão e a imparcialidade das previsões do modelo de ML.

dados em trânsito

Dados que estão se movendo ativamente pela sua rede, como entre os recursos da rede.

data mesh

Um framework de arquitetura que fornece propriedade de dados distribuída e descentralizada com gerenciamento e governança centralizados.

minimização de dados

O princípio de coletar e processar apenas os dados estritamente necessários. Praticar a minimização de dados no Nuvem AWS pode reduzir os riscos de privacidade, os custos e a pegada de carbono de sua análise.

perímetro de dados

Um conjunto de proteções preventivas em seu AWS ambiente que ajudam a garantir que somente identidades confiáveis acessem recursos confiáveis das redes esperadas. Para obter mais informações, consulte [Construindo um perímetro de dados em AWS](#)

pré-processamento de dados

A transformação de dados brutos em um formato que seja facilmente analisado por seu modelo de ML. O pré-processamento de dados pode significar a remoção de determinadas colunas ou linhas e o tratamento de valores ausentes, inconsistentes ou duplicados.

proveniência dos dados

O processo de rastrear a origem e o histórico dos dados ao longo de seu ciclo de vida, por exemplo, como os dados foram gerados, transmitidos e armazenados.

titular dos dados

Um indivíduo cujos dados estão sendo coletados e processados.

data warehouse

Um sistema de gerenciamento de dados compatível com business intelligence, como analytics. Os data warehouses geralmente contêm grandes quantidades de dados históricos e geralmente são usados para consultas e análises.

linguagem de definição de dados (DDL)

Instruções ou comandos para criar ou modificar a estrutura de tabelas e objetos em um banco de dados.

linguagem de manipulação de dados (DML)

Instruções ou comandos para modificar (inserir, atualizar e excluir) informações em um banco de dados.

DDL

Veja [linguagem de definição de banco de dados](#).

deep ensemble

A combinação de vários modelos de aprendizado profundo para gerar previsões. Os deep ensembles podem ser usados para produzir uma previsão mais precisa ou para estimar a incerteza nas previsões.

Aprendizado profundo

Um subcampo do ML que usa várias camadas de redes neurais artificiais para identificar o mapeamento entre os dados de entrada e as variáveis-alvo de interesse.

defense-in-depth

Uma abordagem de segurança da informação na qual uma série de mecanismos e controles de segurança são cuidadosamente distribuídos por toda a rede de computadores para proteger a confidencialidade, a integridade e a disponibilidade da rede e dos dados nela contidos. Ao adotar essa estratégia AWS, você adiciona vários controles em diferentes camadas da AWS Organizations estrutura para ajudar a proteger os recursos. Por exemplo, uma defense-in-depth abordagem pode combinar autenticação multifatorial, segmentação de rede e criptografia.

administrador delegado

Em AWS Organizations, um serviço compatível pode registrar uma conta de AWS membro para administrar as contas da organização e gerenciar as permissões desse serviço. Essa conta é chamada de administrador delegado para esse serviço. Para obter mais informações e uma lista de serviços compatíveis, consulte [Serviços que funcionam com o AWS Organizations](#) na documentação do AWS Organizations .

implantação

O processo de criar uma aplicação, novos recursos ou correções de código disponíveis no ambiente de destino. A implantação envolve a implementação de mudanças em uma base de código e, em seguida, a criação e execução dessa base de código nos ambientes da aplicação

ambiente de desenvolvimento

Veja [ambiente](#).

controle detectivo

Um controle de segurança projetado para detectar, registrar e alertar após a ocorrência de um evento. Esses controles são uma segunda linha de defesa, alertando você sobre eventos de segurança que contornaram os controles preventivos em vigor. Para obter mais informações, consulte [Controles detectivos](#) em Como implementar controles de segurança na AWS.

mapeamento do fluxo de valor de desenvolvimento (DVSM)

Um processo usado para identificar e priorizar restrições que afetam negativamente a velocidade e a qualidade em um ciclo de vida de desenvolvimento de software. O DVSM estende o processo de mapeamento do fluxo de valor originalmente projetado para práticas de manufatura enxuta. Ele se concentra nas etapas e equipes necessárias para criar e movimentar valor por meio do processo de desenvolvimento de software.

gêmeo digital

Uma representação virtual de um sistema real, como um prédio, fábrica, equipamento industrial ou linha de produção. Os gêmeos digitais oferecem suporte à manutenção preditiva, ao monitoramento remoto e à otimização da produção.

tabela de dimensões

Em um [esquema em estrela](#), uma tabela menor que contém atributos de dados sobre dados quantitativos em uma tabela de fatos. Os atributos da tabela de dimensões geralmente são campos de texto ou números discretos que se comportam como texto. Esses atributos normalmente são usados para restringir consultas, filtrar e rotular conjuntos de resultados.

desastre

Um evento que impede que uma workload ou sistema cumpra seus objetivos de negócios em seu local principal de implantação. Esses eventos podem ser desastres naturais, falhas técnicas ou o resultado de ações humanas, como configuração incorreta não intencional ou ataque de malware.

Recuperação de desastres (RD)

A estratégia e o processo que você usa para minimizar o tempo de inatividade e a perda de dados causados por um [desastre](#). Para obter mais informações, consulte [Recuperação de desastres de cargas de trabalho em AWS: Recuperação na nuvem no AWS Well-Architected Framework](#).

DML

Veja [linguagem de manipulação de banco de dados](#).

design orientado por domínio

Uma abordagem ao desenvolvimento de um sistema de software complexo conectando seus componentes aos domínios em evolução, ou principais metas de negócios, atendidos por cada componente. Esse conceito foi introduzido por Eric Evans em seu livro, Design orientado por domínio: lidando com a complexidade no coração do software (Boston: Addison-Wesley Professional, 2003). Para obter informações sobre como usar o design orientado por domínio com o padrão strangler fig, consulte [Modernizar incrementalmente os serviços web herdados do Microsoft ASP.NET \(ASMX\) usando contêineres e o Amazon API Gateway](#).

DR

Veja [recuperação de desastres](#).

Deteção da oscilação

Rastreamento de desvios de uma configuração de linha de base. Por exemplo, você pode usar AWS CloudFormation para [detectar desvios nos recursos do sistema](#) ou AWS Control Tower para [detectar mudanças em seu landing zone](#) que possam afetar a conformidade com os requisitos de governança.

DVSM

Veja [mapeamento do fluxo de valor de desenvolvimento](#).

E

EDA

Veja [análise exploratória de dados](#).

EDI

Veja [intercâmbio eletrônico de dados](#).

computação de borda

A tecnologia que aumenta o poder computacional de dispositivos inteligentes nas bordas de uma rede de IoT. Quando comparada com a [computação em nuvem](#), a computação de borda pode reduzir a latência da comunicação e melhorar o tempo de resposta.

intercâmbio eletrônico de dados (EDI)

A troca automatizada de documentos comerciais entre organizações. Para obter mais informações, consulte [O que é EDI \(Intercâmbio eletrônico de dados\)?](#).

criptografia

Um processo de computação que transforma dados de texto simples, legíveis por humanos, em texto cifrado.

chave de criptografia

Uma sequência criptográfica de bits aleatórios que é gerada por um algoritmo de criptografia. As chaves podem variar em tamanho, e cada chave foi projetada para ser imprevisível e exclusiva.

endianismo

A ordem na qual os bytes são armazenados na memória do computador. Os sistemas big-endian armazenam o byte mais significativo antes. Os sistemas little-endian armazenam o byte menos significativo antes.

endpoint

Veja [endpoint de serviço](#).

serviço de endpoint

Um serviço que pode ser hospedado em uma nuvem privada virtual (VPC) para ser compartilhado com outros usuários. Você pode criar um serviço de endpoint com AWS PrivateLink e conceder permissões a outros diretores Contas da AWS ou a AWS Identity and Access Management (IAM). Essas contas ou entidades principais podem se conectar ao serviço de endpoint de maneira privada criando endpoints da VPC de interface. Para obter mais informações, consulte [Criar um serviço de endpoint](#) na documentação do Amazon Virtual Private Cloud (Amazon VPC).

planejamento de recursos empresariais (ERP)

Um sistema que automatiza e gerencia os principais processos de negócios (como contabilidade, [MES](#) e gerenciamento de projetos) para uma empresa.

criptografia envelopada

O processo de criptografar uma chave de criptografia com outra chave de criptografia. Para obter mais informações, consulte [Criptografia de envelope](#) na documentação AWS Key Management Service (AWS KMS).

ambiente

Uma instância de uma aplicação em execução. Estes são tipos comuns de ambientes na computação em nuvem:

- ambiente de desenvolvimento: uma instância de uma aplicação em execução que está disponível somente para a equipe principal responsável pela manutenção da aplicação. Ambientes de desenvolvimento são usados para testar mudanças antes de promovê-las para ambientes superiores. Esse tipo de ambiente às vezes é chamado de ambiente de teste.
- ambientes inferiores: todos os ambientes de desenvolvimento para uma aplicação, como aqueles usados para compilações e testes iniciais.
- ambiente de produção: uma instância de uma aplicação em execução que os usuários finais podem acessar. Em um CI/CD pipeline, o ambiente de produção é o último ambiente de implantação.
- ambientes superiores: todos os ambientes que podem ser acessados por usuários que não sejam a equipe principal de desenvolvimento. Isso pode incluir um ambiente de produção, ambientes de pré-produção e ambientes para testes de aceitação do usuário.

epic

Em metodologias ágeis, categorias funcionais que ajudam a organizar e priorizar seu trabalho. Os epics fornecem uma descrição de alto nível dos requisitos e das tarefas de implementação. Por exemplo, os épicos de segurança AWS da CAF incluem gerenciamento de identidade e acesso, controles de detetive, segurança de infraestrutura, proteção de dados e resposta a incidentes. Para obter mais informações sobre epics na estratégia de migração da AWS, consulte o [guia de implementação do programa](#).

ERP

Veja [planejamento de recursos empresariais](#).

análise exploratória de dados (EDA)

O processo de analisar um conjunto de dados para entender suas principais características. Você coleta ou agrega dados e, em seguida, realiza investigações iniciais para encontrar padrões, detectar anomalias e verificar suposições. O EDA é realizado por meio do cálculo de estatísticas resumidas e da criação de visualizações de dados.

F

tabela de fatos

A tabela central em um [esquema em estrela](#). Ela armazena dados quantitativos sobre as operações comerciais. Normalmente, uma tabela de fatos contém dois tipos de colunas: as que contêm medidas e as que contêm uma chave externa para uma tabela de dimensões.

Antecipar-se à falha

Uma filosofia que usa testes frequentes e incrementais para reduzir o ciclo de vida do desenvolvimento. É uma parte essencial de uma abordagem ágil.

delimitação de isolamento contra falhas

No Nuvem AWS, um limite, como uma zona de disponibilidade, Região da AWS um plano de controle ou um plano de dados, que limita o efeito de uma falha e ajuda a melhorar a resiliência das cargas de trabalho. Para obter mais informações, consulte [AWS Fault Isolation Boundaries](#).

ramificação de recursos

Veja [ramificação](#).

recursos

Os dados de entrada usados para fazer uma previsão. Por exemplo, em um contexto de manufatura, os recursos podem ser imagens capturadas periodicamente na linha de fabricação.

importância do recurso

O quanto um recurso é importante para as previsões de um modelo. Isso geralmente é expresso como uma pontuação numérica que pode ser calculada por meio de várias técnicas, como Shapley Additive Explanations (SHAP) e gradientes integrados. Para obter mais informações, consulte [Interpretabilidade do modelo de aprendizado de máquina com AWS](#).

transformação de recursos

O processo de otimizar dados para o processo de ML, incluindo enriquecer dados com fontes adicionais, escalar valores ou extrair vários conjuntos de informações de um único campo de dados. Isso permite que o modelo de ML se beneficie dos dados. Por exemplo, se a data “2021-05-27 00:15:37” for dividida em “2021”, “maio”, “quinta” e “15”, isso poderá ajudar o algoritmo de aprendizado a aprender padrões diferenciados associados a diferentes componentes de dados.

prompt few shot

Fornecer a um [LLM](#) um pequeno número de exemplos que demonstram a tarefa e o resultado desejado antes de solicitar que ele execute uma tarefa semelhante. Essa técnica é uma aplicação do aprendizado em contexto, em que os modelos aprendem com exemplos (shots) incorporados aos prompts. Prompts few-shot podem ser eficazes para tarefas que exigem formatação, raciocínio ou conhecimento de domínio específicos. Veja também [prompts zero-shot](#).

FGAC

Veja [controle de acesso refinado](#).

Controle de acesso refinado (FGAC)

O uso de várias condições para permitir ou negar uma solicitação de acesso.

migração flash-cut

Um método de migração de banco de dados que usa replicação contínua de dados via [captura de dados de alteração](#) para migrar os dados no menor tempo possível, em vez de usar uma abordagem em fases. O objetivo é reduzir ao mínimo o tempo de inatividade.

FM

Veja [modelo de base](#).

modelo de base (FM)

Uma grande rede neural de aprendizado profundo que vem treinando em grandes conjuntos de dados generalizados e não rotulados. FMs são capazes de realizar uma ampla variedade de tarefas gerais, como entender a linguagem, gerar texto e imagens e conversar em linguagem natural. Para obter mais informações, consulte [O que são modelos de base?](#).

G

IA generativa

Um subconjunto de modelos de [IA](#) que foram treinados em grandes quantidades de dados e que podem usar um simples prompt de texto para criar novos artefatos e conteúdo, como imagens, vídeos, texto e áudio. Para obter mais informações, consulte [O que é IA generativa?](#).

bloqueio geográfico

Veja [restrições geográficas](#).

restrições geográficas (bloqueio geográfico)

Na Amazon CloudFront, uma opção para impedir que usuários em países específicos acessem distribuições de conteúdo. É possível usar uma lista de permissões ou uma lista de bloqueios para especificar países aprovados e banidos. Para obter mais informações, consulte [Restringir a distribuição geográfica do seu conteúdo](#) na CloudFront documentação.

Fluxo de trabalho do GitFlow

Uma abordagem na qual ambientes inferiores e superiores usam ramificações diferentes em um repositório de código-fonte. O fluxo de trabalho do Gitflow é considerado legado, e o [fluxo de trabalho trunk-based](#) é a abordagem moderna e preferencial.

golden image

Um snapshot de um sistema ou software usado como modelo para implantar novas instâncias desse sistema ou software. Por exemplo, na manufatura, uma golden image pode ser usada para provisionar software em vários dispositivos e ajudar a melhorar a velocidade, a escalabilidade e a produtividade nas operações de fabricação de dispositivos.

estratégia greenfield

A ausência de infraestrutura existente em um novo ambiente. Ao adotar uma estratégia greenfield para uma arquitetura de sistema, é possível selecionar todas as novas tecnologias sem a restrição da compatibilidade com a infraestrutura existente, também conhecida como [brownfield](#). Se estiver expandindo a infraestrutura existente, poderá combinar as estratégias brownfield e greenfield.

barreira de proteção

Uma regra de alto nível que ajuda a governar recursos, políticas e conformidade em todas as unidades organizacionais (OUs). Barreiras de proteção preventivas impõem políticas para garantir o alinhamento a padrões de conformidade. Elas são implementadas usando políticas de controle de serviço e limites de permissões do IAM. Barreiras de proteção detectivas detectam violações de políticas e problemas de conformidade e geram alertas para remediação. Eles são implementados usando AWS Config, AWS Security Hub CSPM, Amazon GuardDuty AWS Trusted Advisor, Amazon Inspector e verificações personalizadas AWS Lambda .

H

HA

Veja [alta disponibilidade](#).

migração heterogênea de bancos de dados

Migrar seu banco de dados de origem para um banco de dados de destino que usa um mecanismo de banco de dados diferente (por exemplo, Oracle para Amazon Aurora). A migração heterogênea geralmente faz parte de um esforço de redefinição da arquitetura, e converter

o esquema pode ser uma tarefa complexa. [O AWS fornece o AWS SCT](#) para ajudar nas conversões de esquemas.

alta disponibilidade (HA)

A capacidade de uma workload operar continuamente, sem intervenção, em caso de desafios ou desastres. Os sistemas HA são projetados para realizar o failover automático, oferecer consistentemente desempenho de alta qualidade e lidar com diferentes cargas e falhas com impacto mínimo no desempenho.

modernização de historiador

Uma abordagem usada para modernizar e atualizar os sistemas de tecnologia operacional (OT) para melhor atender às necessidades do setor de manufatura. Um historiador é um tipo de banco de dados usado para coletar e armazenar dados de várias fontes em uma fábrica.

dados de hold-out

Uma parte dos dados históricos rotulados que são retidos de um conjunto de dados usado para treinar um modelo de [machine learning](#). Você pode usar dados de hold-out para avaliar a performance do modelo comparando as previsões do modelo com os dados de retenção.

migração homogênea de bancos de dados

Migrar seu banco de dados de origem para um banco de dados de destino que compartilha o mesmo mecanismo de banco de dados (por exemplo, Microsoft SQL Server para Amazon RDS para SQL Server). A migração homogênea geralmente faz parte de um esforço de redefinição da hospedagem ou da plataforma. É possível usar utilitários de banco de dados nativos para migrar o esquema.

dados quentes

Dados acessados com frequência, como dados em tempo real ou dados translacionais recentes. Esses dados normalmente exigem uma camada ou classe de armazenamento de alto desempenho para fornecer respostas rápidas às consultas.

hotfix

Uma correção urgente para um problema crítico em um ambiente de produção. Devido à sua urgência, um hotfix geralmente é feito fora do fluxo de trabalho normal de DevOps lançamento.

período de hipercuidados

Imediatamente após a substituição, o período em que uma equipe de migração gerencia e monitora as aplicações migradas na nuvem para resolver quaisquer problemas. Normalmente,

a duração desse período é de 1 a 4 dias. No final do período de hipercuidados, a equipe de migração normalmente transfere a responsabilidade pelas aplicações para a equipe de operações de nuvem.

eu

IaC

Veja [infraestrutura como código](#).

Política baseada em identidade

Uma política anexada a um ou mais diretores do IAM que define suas permissões no Nuvem AWS ambiente.

aplicação ociosa

Uma aplicação que tem um uso médio de CPU e memória entre 5 e 20% em um período de 90 dias. Em um projeto de migração, é comum retirar essas aplicações ou retê-las on-premises.

IIoT

Veja [Internet das Coisas Industrial](#).

infraestrutura imutável

Um modelo que implanta uma nova infraestrutura para workloads de produção em vez de atualizar, aplicar patches ou modificar a infraestrutura existente. Infraestruturas imutáveis são inerentemente mais consistentes, confiáveis e preditivas do que [infraestruturas mutáveis](#). Para obter mais informações, consulte a prática recomendada [Implantar usando infraestrutura imutável](#) no AWS Well-Architected Framework.

VPC de entrada (admissão)

Em uma arquitetura de AWS várias contas, uma VPC que aceita, inspeciona e roteia conexões de rede de fora de um aplicativo. A [Arquitetura de Referência de AWS Segurança](#) recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

migração incremental

Uma estratégia de substituição na qual você migra a aplicação em pequenas partes, em vez de realizar uma única substituição completa. Por exemplo, é possível mover inicialmente

apenas alguns microsserviços ou usuários para o novo sistema. Depois de verificar se tudo está funcionando corretamente, mova os microsserviços ou usuários adicionais de forma incremental até poder descomissionar seu sistema herdado. Essa estratégia reduz os riscos associados a migrações de grande porte.

Indústria 4.0

Um termo que foi introduzido por [Klaus Schwab](#) em 2016 para se referir à modernização dos processos de manufatura por meio de avanços em conectividade, dados em tempo real, automação, analytics e IA/ML.

infraestrutura

Todos os recursos e ativos contidos no ambiente de uma aplicação.

Infraestrutura como código (IaC)

O processo de provisionamento e gerenciamento da infraestrutura de uma aplicação por meio de um conjunto de arquivos de configuração. A IaC foi projetada para ajudar você a centralizar o gerenciamento da infraestrutura, padronizar recursos e escalar rapidamente para que novos ambientes sejam reproduzíveis, confiáveis e consistentes.

Internet industrial das coisas (IIoT)

O uso de sensores e dispositivos conectados à Internet nos setores industriais, como manufatura, energia, automotivo, saúde, ciências biológicas e agricultura. Para obter mais informações, consulte [Criando uma estratégia de transformação digital industrial da Internet das Coisas \(IIoT\)](#).

VPC de inspeção

Em uma arquitetura de AWS várias contas, uma VPC centralizada que gerencia as inspeções do tráfego de rede entre VPCs (na mesma ou em diferentes Regiões da AWS) a Internet e as redes locais. A [Arquitetura de Referência de AWS Segurança](#) recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

Internet das coisas (IoT)

A rede de objetos físicos conectados com sensores ou processadores incorporados que se comunicam com outros dispositivos e sistemas pela Internet ou por uma rede de comunicação local. Para obter mais informações, consulte [O que é IoT?](#)

interpretabilidade

Uma característica de um modelo de machine learning que descreve o grau em que um ser humano pode entender como as previsões do modelo dependem de suas entradas. Para obter mais informações, consulte [Interpretabilidade do modelo de aprendizado de máquina com AWS](#).

IoT

Veja [Internet das Coisas](#).

Biblioteca de informações de TI (ITIL)

Um conjunto de práticas recomendadas para fornecer serviços de TI e alinhar esses serviços a requisitos de negócios. A ITIL fornece a base para o ITSM.

Gerenciamento de serviços de TI (ITSM)

Atividades associadas a design, implementação, gerenciamento e suporte de serviços de TI para uma organização. Para obter informações sobre a integração de operações em nuvem com ferramentas de ITSM, consulte o [guia de integração de operações](#).

ITIL

Veja [biblioteca de informações de TI](#).

ITSM

Veja [gerenciamento de serviços de TI](#).

L

controle de acesso baseado em etiqueta (LBAC)

Uma implementação do controle de acesso obrigatório (MAC) em que os usuários e os dados em si recebem explicitamente um valor de etiqueta de segurança. A interseção entre a etiqueta de segurança do usuário e a etiqueta de segurança dos dados determina quais linhas e colunas podem ser vistas pelo usuário.

zona de pouso

Uma landing zone é um AWS ambiente bem arquitetado, com várias contas, escalável e seguro. Um ponto a partir do qual suas organizações podem iniciar e implantar rapidamente workloads e aplicações com confiança em seu ambiente de segurança e infraestrutura. Para obter mais

informações sobre zonas de pouso, consulte [Configurar um ambiente da AWS com várias contas seguro e escalável](#).

grande modelo de linguagem (LLM)

Um modelo de [IA](#) de aprendizado profundo pré-treinado em uma grande quantidade de dados. Um LLM pode realizar várias tarefas, como responder a perguntas, resumir documentos, traduzir texto para outros idiomas e completar frases. Para obter mais informações, consulte [O que são LLMs](#).

migração de grande porte

Uma migração de 300 servidores ou mais.

LBAC

Veja [controle de acesso baseado em rótulo](#).

privilegio mínimo

A prática recomendada de segurança de conceder as permissões mínimas necessárias para executar uma tarefa. Para obter mais informações, consulte [Aplicar permissões de privilégios mínimos](#) na documentação do IAM.

mover sem alterações (lift-and-shift)

Veja [7 Rs](#).

sistema little-endian

Um sistema que armazena o byte menos significativo antes. Veja também [endianness](#).

LLM

Veja [grande modelo de linguagem](#).

ambientes inferiores

Veja [ambiente](#).

M

machine learning (ML)

Um tipo de inteligência artificial que usa algoritmos e técnicas para reconhecimento e aprendizado de padrões. O ML analisa e aprende com dados gravados, por exemplo, dados da

Internet das Coisas (IoT), para gerar um modelo estatístico baseado em padrões. Para obter mais informações, consulte [Machine learning](#).

ramificação principal

Veja [ramificação](#).

Malware

Software projetado para comprometer a segurança ou a privacidade do computador. O malware pode interromper os sistemas do computador, vaziar informações sensíveis ou obter acesso não autorizado. Exemplos de malware incluem vírus, worms, ransomware, cavalos de Troia, spyware e keyloggers.

Serviços gerenciados

Serviços da AWS para o qual AWS opera a camada de infraestrutura, o sistema operacional e as plataformas, e você acessa os endpoints para armazenar e recuperar dados. O Amazon Simple Storage Service (Amazon S3) e o Amazon DynamoDB são exemplos de serviços gerenciados. Eles também são conhecidos como serviços abstraídos.

sistema de execução de manufatura (MES)

Um sistema de software para rastrear, monitorar, documentar e controlar processos de produção que convertem matérias-primas em produtos acabados no chão de fábrica.

MAP

Veja [Programa de Aceleração da Migração](#).

mecanismo

Um processo completo em que você cria uma ferramenta, impulsiona a adoção da ferramenta e, em seguida, inspeciona os resultados para fazer ajustes. Um mecanismo é um ciclo que se reforça e se aprimora à medida que opera. Para obter mais informações, consulte [Construindo mecanismos](#) no AWS Well-Architected Framework.

conta de membro

Todos, Contas da AWS exceto a conta de gerenciamento, que fazem parte de uma organização em AWS Organizations. Uma conta só pode ser membro de uma organização de cada vez.

MES

Veja [sistema de execução de manufatura](#).

Transporte de Telemetria de Enfileiramento de Mensagens (MQTT)

[Um protocolo de comunicação leve machine-to-machine \(M2M\), baseado no padrão de publicação/assinatura, para dispositivos de IoT com recursos limitados.](#)

microserviço

Um serviço pequeno e independente que se comunica de forma bem definida APIs e normalmente é de propriedade de equipes pequenas e independentes. Por exemplo, um sistema de seguradora pode incluir microserviços que mapeiam as capacidades comerciais, como vendas ou marketing, ou subdomínios, como compras, reclamações ou análises. Os benefícios dos microserviços incluem agilidade, escalabilidade flexível, fácil implantação, código reutilizável e resiliência. Para obter mais informações, consulte [Integração de microserviços usando serviços sem AWS servidor](#).

arquitetura de microserviços

Uma abordagem à criação de aplicações com componentes independentes que executam cada processo de aplicação como um microserviço. Esses microserviços se comunicam por meio de uma interface bem definida usando leveza. APIs Cada microserviço nessa arquitetura pode ser atualizado, implantado e escalado para atender à demanda por funções específicas de uma aplicação. Para obter mais informações, consulte [Implementação de microserviços em. AWS](#)

Programa de Aceleração da Migração (MAP)

Um AWS programa que fornece suporte de consultoria, treinamento e serviços para ajudar as organizações a criar uma base operacional sólida para migrar para a nuvem e ajudar a compensar o custo inicial das migrações. O MAP inclui uma metodologia de migração para executar migrações legadas de forma metódica e um conjunto de ferramentas para automatizar e acelerar cenários comuns de migração.

migração em escala

O processo de mover a maior parte do portfólio de aplicações para a nuvem em ondas, com mais aplicações sendo movidas em um ritmo mais rápido a cada onda. Essa fase usa as práticas recomendadas e lições aprendidas nas fases anteriores para implementar uma fábrica de migração de equipes, ferramentas e processos para agilizar a migração de workloads por meio de automação e entrega ágeis. Esta é a terceira fase da [estratégia de migração para a AWS](#).

fábrica de migração

Equipes multifuncionais que simplificam a migração de workloads por meio de abordagens automatizadas e ágeis. As equipes da fábrica de migração geralmente incluem operações,

analistas e proprietários de negócios, engenheiros de migração, desenvolvedores e DevOps profissionais que trabalham em sprints. Entre 20 e 50% de um portfólio de aplicações corporativas consiste em padrões repetidos que podem ser otimizados por meio de uma abordagem de fábrica. Para obter mais informações, consulte [discussão sobre fábricas de migração](#) e o [guia do Cloud Migration Factory](#) neste conjunto de conteúdo.

metadados de migração

As informações sobre a aplicação e o servidor necessárias para concluir a migração. Cada padrão de migração exige um conjunto de metadados de migração diferente. Exemplos de metadados de migração incluem a sub-rede, o grupo de segurança e AWS a conta de destino.

padrão de migração

Uma tarefa de migração repetível que detalha a estratégia de migração, o destino da migração e a aplicação ou o serviço de migração usado. Exemplo: rehospede a migração para o Amazon EC2 AWS com o Application Migration Service.

Avaliação de Portfólio para Migração (MPA)

Uma ferramenta on-line que fornece informações para validar o caso de negócios para migrar para a Nuvem AWS. O MPA fornece avaliação detalhada do portfólio (dimensionamento correto do servidor, preços, comparações de TCO, análise de custos de migração), bem como planejamento de migração (análise e coleta de dados de aplicações, agrupamento de aplicações, priorização de migração e planejamento de ondas). A [ferramenta MPA](#) (requer login) está disponível gratuitamente para todos os AWS consultores e consultores parceiros da APN.

Avaliação de Preparação para Migração (MRA)

O processo de obter insights sobre o status de prontidão de uma organização para a nuvem, identificar pontos fortes e fracos e criar um plano de ação para fechar as lacunas identificadas, usando o CAF. AWS Para mais informações, consulte o [guia de preparação para migração](#). A MRA é a primeira fase da [estratégia de migração para a AWS](#).

estratégia de migração

A abordagem usada para migrar uma workload para a Nuvem AWS. Para obter mais informações, veja a entrada [7 Rs](#) neste glossário e consulte [Mobilize sua organização para acelerar migrações em grande escala](#).

ML

Veja [machine learning](#).

modernização

Transformar uma aplicação desatualizada (herdada ou monolítica) e sua infraestrutura em um sistema ágil, elástico e altamente disponível na nuvem para reduzir custos, ganhar eficiência e aproveitar as inovações. Para obter mais informações, consulte [Strategy for modernizing applications in the Nuvem AWS](#).

avaliação de preparação para modernização

Uma avaliação que ajuda a determinar a preparação para modernização das aplicações de uma organização. Ela identifica benefícios, riscos e dependências e determina o quão bem a organização pode acomodar o estado futuro dessas aplicações. O resultado da avaliação é um esquema da arquitetura de destino, um roteiro que detalha as fases de desenvolvimento e os marcos do processo de modernização e um plano de ação para abordar as lacunas identificadas. Para obter mais informações, consulte [Evaluating modernization readiness for applications in the Nuvem AWS](#).

aplicações monolíticas (monólitos)

Aplicações que são executadas como um único serviço com processos fortemente acoplados. As aplicações monolíticas apresentam várias desvantagens. Se um recurso da aplicação apresentar um aumento na demanda, toda a arquitetura deverá ser escalada. Adicionar ou melhorar os recursos de uma aplicação monolítica também se torna mais complexo quando a base de código cresce. Para resolver esses problemas, é possível criar uma arquitetura de microsserviços. Para obter mais informações, consulte [Decompor monólitos em microsserviços](#).

MPA

Veja [Avaliação do Portfólio para Migração](#).

MQTT

Veja [Transporte de Telemetria de Enfileiramento de Mensagens](#).

classificação multiclasse

Um processo que ajuda a gerar previsões para várias classes (prevendo um ou mais de dois resultados). Por exemplo, um modelo de ML pode perguntar “Este produto é um livro, um carro ou um telefone?” ou “Qual categoria de produtos é mais interessante para este cliente?”

infraestrutura mutável

Um modelo que atualiza e modifica a infraestrutura existente para workloads de produção. Para melhorar a consistência, confiabilidade e previsibilidade, o AWS Well-Architected Framework recomenda o uso de infraestrutura [imutável](#) como uma prática recomendada.

O

OAC

Veja [controle de acesso de origem](#).

OAI

Veja [identidade de acesso de origem](#).

OCM

Veja [gerenciamento de alterações organizacionais](#).

migração offline

Um método de migração no qual a workload de origem é desativada durante o processo de migração. Esse método envolve tempo de inatividade prolongado e geralmente é usado para workloads pequenas e não críticas.

OI

Veja [integração de operações](#).

Ola

Veja [acordo de nível operacional](#).

migração online

Um método de migração no qual a workload de origem é copiada para o sistema de destino sem ser colocada offline. As aplicações conectadas à workload podem continuar funcionando durante a migração. Esse método envolve um tempo de inatividade nulo ou mínimo e normalmente é usado para workloads essenciais para a produção.

OPC-UA

Veja [Open Process Communications - Unified Architecture](#).

Open Process Communications - Unified Architecture (OPC-UA)

Um protocolo de comunicação machine-to-machine (M2M) para automação industrial. O OPC-UA fornece um padrão de interoperabilidade com esquemas de criptografia, autenticação e autorização de dados.

acordo de nível operacional (OLA)

Um acordo que esclarece o que os grupos funcionais de TI prometem oferecer uns aos outros para apoiar um acordo de serviço (SLA).

análise de prontidão operacional (ORR)

Uma lista de verificação de perguntas e práticas recomendadas associadas que ajudam você a entender, avaliar, prevenir ou reduzir o escopo de incidentes e possíveis falhas. Para obter mais informações, consulte [Operational Readiness Reviews \(ORR\)](#) no AWS Well-Architected Framework.

tecnologia operacional (TO)

Sistemas de hardware e software que trabalham com o ambiente físico para controlar operações, equipamentos e infraestrutura industriais. Na manufatura, a integração dos sistemas de tecnologia da informação (TI) e tecnologia operacional (TO) é o foco principal das transformações da [Indústria 4.0](#).

integração de operações (OI)

O processo de modernização das operações na nuvem, que envolve planejamento de preparação, automação e integração. Para obter mais informações, consulte o [guia de integração de operações](#).

trilha organizacional

Uma trilha criada por ela AWS CloudTrail registra todos os eventos de todas as Contas da AWS em uma organização em AWS Organizations. Essa trilha é criada em cada Conta da AWS que faz parte da organização e monitora a atividade em cada conta. Para obter mais informações, consulte [Criação de uma trilha para uma organização](#) na CloudTrail documentação.

gerenciamento de alterações organizacionais (OCM)

Uma estrutura para gerenciar grandes transformações de negócios disruptivas de uma perspectiva de pessoas, cultura e liderança. O OCM ajuda as organizações a se prepararem e fazerem a transição para novos sistemas e estratégias, acelerando a adoção de alterações, abordando questões de transição e promovendo mudanças culturais e organizacionais. Na estratégia de AWS migração, essa estrutura é chamada de aceleração de pessoas, devido à velocidade de mudança exigida nos projetos de adoção da nuvem. Para obter mais informações, consulte o [guia do OCM](#).

controle de acesso de origem (OAC)

Em CloudFront, uma opção aprimorada para restringir o acesso para proteger seu conteúdo do Amazon Simple Storage Service (Amazon S3). O OAC oferece suporte a todos os buckets S3 Regiões da AWS, criptografia do lado do servidor com AWS KMS (SSE-KMS) e solicitações dinâmicas ao bucket S3. PUT DELETE

Identidade do acesso de origem (OAI)

Em CloudFront, uma opção para restringir o acesso para proteger seu conteúdo do Amazon S3. Quando você usa o OAI, CloudFront cria um principal com o qual o Amazon S3 pode se autenticar. Os diretores autenticados podem acessar o conteúdo em um bucket do S3 somente por meio de uma distribuição específica. CloudFront Veja também [OAC](#), que fornece um controle de acesso mais granular e aprimorado.

ORR

Veja [análise de prontidão operacional](#).

OT

Veja [tecnologia operacional](#).

VPC de saída (egresso)

Em uma arquitetura de AWS várias contas, uma VPC que gerencia conexões de rede que são iniciadas de dentro de um aplicativo. A [Arquitetura de Referência de AWS Segurança](#) recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

P

limite de permissões

Uma política de gerenciamento do IAM anexada a entidades principais do IAM para definir as permissões máximas que o usuário ou perfil podem ter. Para obter mais informações, consulte [Limites de permissões](#) na documentação do IAM.

Informações de identificação pessoal (PII)

Informações que, quando visualizadas diretamente ou combinadas com outros dados relacionados, podem ser usadas para inferir razoavelmente a identidade de um indivíduo. Exemplos de PII incluem nomes, endereços e informações de contato.

PII

Veja [informações de identificação pessoal](#).

manual

Um conjunto de etapas predefinidas que capturam o trabalho associado às migrações, como a entrega das principais funções operacionais na nuvem. Um manual pode assumir a forma de scripts, runbooks automatizados ou um resumo dos processos ou etapas necessários para operar seu ambiente modernizado.

PLC

Veja [controlador lógico programável](#).

PLM

Veja [gerenciamento do ciclo de vida do produto](#).

política

Um objeto que pode definir permissões (veja [política baseada em identidade](#)), especificar condições de acesso (veja [política baseada em recurso](#)) ou definir as permissões máximas para todas as contas em uma organização no AWS Organizations (veja [política de controle de serviços](#)).

persistência poliglota

Escolher de forma independente a tecnologia de armazenamento de dados de um microsserviço com base em padrões de acesso a dados e outros requisitos. Se seus microsserviços tiverem a mesma tecnologia de armazenamento de dados, eles poderão enfrentar desafios de implementação ou apresentar baixa performance. Os microsserviços serão implementados com mais facilidade e alcançarão performance e escalabilidade melhores se usarem o armazenamento de dados mais bem adaptado às suas necessidades.

avaliação do portfólio

Um processo de descobrir, analisar e priorizar o portfólio de aplicações para planejar a migração. Para obter mais informações, consulte [Avaliar a preparação para a migração](#).

predicado

Uma condição de consulta que retorna `true` ou `false`, normalmente localizada em uma cláusula `WHERE`.

pushdown de predicados

Uma técnica de otimização de consultas de banco de dados que filtra os dados na consulta antes da transferência. Isso reduz a quantidade de dados que devem ser recuperados e processados do banco de dados relacional e melhora a performance das consultas.

controle preventivo

Um controle de segurança projetado para evitar que um evento ocorra. Esses controles são a primeira linha de defesa para ajudar a evitar acesso não autorizado ou alterações indesejadas em sua rede. Para obter mais informações, consulte [Controles preventivos](#) em Como implementar controles de segurança na AWS.

principal (entidade principal)

Uma entidade AWS que pode realizar ações e acessar recursos. Essa entidade geralmente é um usuário raiz para um Conta da AWS, uma função do IAM ou um usuário. Para obter mais informações, consulte Entidade principal em [Termos e conceitos de perfis](#) na documentação do IAM.

Privacidade por design

Uma abordagem em engenharia de sistemas que leva em consideração a privacidade em todo o processo de desenvolvimento.

zonas hospedadas privadas

Um contêiner que contém informações sobre como você deseja que o Amazon Route 53 responda às consultas de DNS para um domínio e seus subdomínios em um ou mais VPCs. Para obter mais informações, consulte [Como trabalhar com zonas hospedadas privadas](#) na documentação do Route 53.

controle proativo

Um [controle de segurança](#) desenvolvido para evitar a implantação de recursos não conformes. Esses controles verificam os recursos antes de serem provisionados. Se o recurso não estiver em conformidade com o controle, ele não será provisionado. Para obter mais informações, consulte o [guia de referência de controles](#) na AWS Control Tower documentação e consulte [Controles proativos](#) em Implementação de controles de segurança em AWS.

gerenciamento do ciclo de vida do produto (PLM)

O gerenciamento de dados e processos de um produto em todo o seu ciclo de vida, desde a concepção, o desenvolvimento e o lançamento, passando pelo crescimento e maturidade, até o declínio e a remoção.

ambiente de produção

Veja [ambiente](#).

controlador lógico programável (PLC)

Na manufatura, um computador altamente confiável e adaptável que monitora as máquinas e automatiza os processos de fabricação.

encadeamento de prompts

Uso da saída de um prompt do [LLM](#) como entrada para o próximo prompt para gerar respostas melhores. Essa técnica é usada para dividir uma tarefa complexa em subtarefas, ou para refinar ou expandir iterativamente uma resposta preliminar. Isso ajuda a melhorar a precisão e a relevância das respostas de um modelo e permite resultados mais granulares e personalizados.

pseudonimização

O processo de substituir identificadores pessoais em um conjunto de dados por valores de espaço reservado. A pseudonimização pode ajudar a proteger a privacidade pessoal. Os dados pseudonimizados ainda são considerados dados pessoais.

publish/subscribe (pub/sub)

Um padrão que permite comunicações assíncronas entre microsserviços para melhorar a escalabilidade e a capacidade de resposta. Por exemplo, em um [MES](#) baseado em microsserviços, um microsserviço pode publicar mensagens de eventos em um canal em que outros microsserviços possam assinar. O sistema pode adicionar novos microsserviços sem alterar o serviço de publicação.

Q

plano de consulta

Uma série de etapas, como instruções, usadas para acessar os dados em um sistema de banco de dados relacional SQL.

regressão de planos de consultas

Quando um otimizador de serviço de banco de dados escolhe um plano menos adequado do que escolhia antes de uma determinada alteração no ambiente de banco de dados ocorrer. Isso pode ser causado por alterações em estatísticas, restrições, configurações do ambiente, associações de parâmetros de consulta e atualizações do mecanismo de banco de dados.

R

Matriz RACI

Veja [responsável, aprovador, consultado, informado \(RACI\)](#).

RAG

Veja [geração aumentada via recuperação](#).

ransomware

Um software mal-intencionado desenvolvido para bloquear o acesso a um sistema ou dados de computador até que um pagamento seja feito.

Matriz RASCI

Veja [responsável, aprovador, consultado, informado \(RACI\)](#).

RCAC

Veja [controle de acesso por linha e coluna](#).

réplica de leitura

Uma cópia de um banco de dados usada somente para leitura. É possível encaminhar consultas para a réplica de leitura e reduzir a carga no banco de dados principal.

Redefinir arquitetura

Veja [7 Rs](#).

objetivo de ponto de recuperação (RPO).

O máximo período de tempo aceitável desde o último ponto de recuperação de dados.

Isso determina o que é considerado uma perda aceitável de dados entre o último ponto de recuperação e a interrupção do serviço.

objetivo de tempo de recuperação (RTO)

O máximo atraso aceitável entre a interrupção e a restauração do serviço.

refatorar

Veja [7 Rs](#).

Região

Uma coleção de AWS recursos em uma área geográfica. Cada um Região da AWS é isolado e independente dos outros para fornecer tolerância a falhas, estabilidade e resiliência. Para obter informações, consulte [Specify which Regiões da AWS your account can use](#).

regressão

Uma técnica de ML que prevê um valor numérico. Por exemplo, para resolver o problema de “Por qual preço esta casa será vendida?” um modelo de ML pode usar um modelo de regressão linear para prever o preço de venda de uma casa com base em fatos conhecidos sobre a casa (por exemplo, a metragem quadrada).

redefinir a hospedagem

Veja [7 Rs](#).

versão

Em um processo de implantação, o ato de promover mudanças em um ambiente de produção.

realocar

Veja [7 Rs](#).

redefinir a plataforma

Veja [7 Rs](#).

recomprar

Veja [7 Rs](#).

resiliência

A capacidade de uma aplicação de resistir ou se recuperar de interrupções. [Alta disponibilidade](#) e [recuperação de desastres](#) são considerações comuns ao planejar a resiliência na Nuvem AWS. Para obter mais informações, consulte [Nuvem AWS Resilience](#).

política baseada em recurso

Uma política associada a um recurso, como um bucket do Amazon S3, um endpoint ou uma chave de criptografia. Esse tipo de política especifica quais entidades principais têm acesso permitido, ações válidas e quaisquer outras condições que devem ser atendidas.

matriz responsável, accountable, consultada, informada (RACI)

Uma matriz que define as funções e responsabilidades de todas as partes envolvidas nas atividades de migração e nas operações de nuvem. O nome da matriz é derivado dos tipos de responsabilidade definidos na matriz: responsável (R), responsabilizável (A), consultado (C) e informado (I). O tipo de suporte (S) é opcional. Se você incluir suporte, a matriz será chamada de matriz RASCI e, se excluir, será chamada de matriz RACI.

controle responsivo

Um controle de segurança desenvolvido para conduzir a remediação de eventos adversos ou desvios em relação à linha de base de segurança. Para obter mais informações, consulte [Controles responsivos](#) em Como implementar controles de segurança na AWS.

reter

Veja [7 Rs](#).

Retirada

Veja [7 Rs](#).

Geração Aumentada de Recuperação (RAG)

Uma tecnologia de [IA generativa](#) em que um [LLM](#) faz referência a uma fonte de dados autorizada que está fora de suas fontes de dados de treinamento antes de gerar uma resposta. Por exemplo, um modelo RAG pode realizar uma pesquisa semântica na base de conhecimento ou nos dados personalizados de uma organização. Para obter mais informações, consulte [O que é RAG \(geração aumentada via recuperação\)?](#).

alternância

O processo de atualizar periodicamente um [segredo](#) para dificultar o acesso de um invasor às credenciais.

controle de acesso por linha e coluna (RCAC)

O uso de expressões SQL básicas e flexíveis que tenham regras de acesso definidas. O RCAC consiste em permissões de linha e máscaras de coluna.

RPO

Veja [objetivo de ponto de recuperação](#).

RTO

Veja [objetivo de tempo de recuperação](#).

runbook

Um conjunto de procedimentos manuais ou automatizados necessários para realizar uma tarefa específica. Eles são normalmente criados para agilizar operações ou procedimentos repetitivos com altas taxas de erro.

S

SAML 2.0

Um padrão aberto que muitos provedores de identidade (IdPs) usam. Esse recurso permite o login único federado (SSO), para que os usuários possam fazer login no Console de gerenciamento da AWS ou chamar as operações da AWS API sem que você precise criar um usuário no IAM para todos em sua organização. Para obter mais informações sobre a federação baseada em SAML 2.0, consulte [Sobre a federação baseada em SAML 2.0](#) na documentação do IAM.

SCADA

Veja [controle de supervisão e aquisição de dados](#).

SCP

Veja [política de controle de serviço](#).

secret

Em AWS Secrets Manager, informações confidenciais ou restritas, como uma senha ou credenciais de usuário, que você armazena de forma criptografada. Consiste no valor secreto e em seus metadados. O valor secreto pode ser binário, uma única string ou várias strings. Para obter mais informações, consulte [What's in a Secrets Manager secret?](#) na documentação do Secrets Manager.

segurança desde a concepção

Uma abordagem em engenharia de sistemas que leva em consideração a segurança em todo o processo de desenvolvimento.

controle de segurança

Uma barreira de proteção técnica ou administrativa que impede, detecta ou reduz a capacidade de uma ameaça explorar uma vulnerabilidade de segurança. Existem quatro tipos primários de controles de segurança: [preventivos](#), [detectivos](#), [responsivos](#) e [proativos](#).

hardening da segurança

O processo de reduzir a superfície de ataque para torná-la mais resistente a ataques. Isso pode incluir ações como remover recursos que não são mais necessários, implementar a prática recomendada de segurança de conceder privilégios mínimos ou desativar recursos desnecessários em arquivos de configuração.

sistema de gerenciamento de eventos e informações de segurança (SIEM)

Ferramentas e serviços que combinam sistemas de gerenciamento de informações de segurança (SIM) e gerenciamento de eventos de segurança (SEM). Um sistema SIEM coleta, monitora e analisa dados de servidores, redes, dispositivos e outras fontes para detectar ameaças e violações de segurança e gerar alertas.

automação de resposta de segurança

Uma ação predefinida e programada projetada para responder ou remediar automaticamente um evento de segurança. Essas automações servem como controles de segurança [responsivos](#) ou [detectivos](#) que ajudam você a implementar as melhores práticas AWS de segurança. Exemplos de ações de resposta automatizada incluem a modificação de um grupo de segurança da VPC, a aplicação de patches em uma instância do Amazon EC2 ou a alternância de credenciais.

Criptografia do lado do servidor

Criptografia dos dados em seu destino, por AWS service (Serviço da AWS) quem os recebe.

política de controle de serviços (SCP)

Uma política que fornece controle centralizado sobre as permissões de todas as contas em uma organização em AWS Organizations. SCPs defina barreiras ou estabeleça limites nas ações que um administrador pode delegar a usuários ou funções. Você pode usar SCPs como listas de permissão ou listas de negação para especificar quais serviços ou ações são permitidos ou proibidos. Para obter mais informações, consulte [Políticas de controle de serviço](#) na AWS Organizations documentação.

service endpoint (endpoint de serviço)

O URL do ponto de entrada para um AWS service (Serviço da AWS). Você pode usar o endpoint para se conectar programaticamente ao serviço de destino. Para obter mais informações, consulte [Endpoints do AWS service \(Serviço da AWS\)](#) na Referência geral da AWS.

acordo de serviço (SLA)

Um acordo que esclarece o que uma equipe de TI promete fornecer aos clientes, como tempo de atividade e performance do serviço.

indicador de nível de serviço (SLI)

Uma avaliação de um aspecto de performance de um serviço, como taxa de erro, disponibilidade ou throughput.

objetivo de nível de serviço (SLO)

Uma métrica alvo que representa a integridade de um serviço, conforme avaliado por um [indicador de nível de serviço](#).

modelo de responsabilidade compartilhada

Um modelo que descreve a responsabilidade com a qual você compartilha AWS pela segurança e conformidade na nuvem. AWS é responsável pela segurança da nuvem, enquanto você é responsável pela segurança na nuvem. Para obter mais informações, consulte o [Modelo de responsabilidade compartilhada](#).

SIEM

Veja [sistema de gerenciamento de eventos e informações de segurança](#).

ponto único de falha (SPOF)

Uma falha em um único componente crítico de uma aplicação que pode interromper o sistema.

SLA

Veja [acordo de serviço](#).

SLI

Veja [indicador de nível de serviço](#).

SLO

Veja [objetivo de nível de serviço](#).

split-and-seed modelo

Um padrão para escalar e acelerar projetos de modernização. À medida que novos recursos e lançamentos de produtos são definidos, a equipe principal se divide para criar novas equipes de produtos. Isso ajuda a escalar os recursos e os serviços da sua organização, melhora a produtividade do desenvolvedor e possibilita inovações rápidas. Para obter mais informações, consulte [Phased approach to modernizing applications in the Nuvem AWS](#).

SPOF

Veja [ponto único de falha](#).

esquema em estrela

Uma estrutura organizacional de banco de dados que usa uma grande tabela de fatos para armazenar dados transacionais ou medidos e usa uma ou mais tabelas dimensionais menores para armazenar atributos de dados. Essa estrutura foi projetada para ser usada em um [data warehouse](#) ou para fins de inteligência comercial.

padrão strangler fig

Uma abordagem à modernização de sistemas monolíticos que consiste em reescrever e substituir incrementalmente a funcionalidade do sistema até que o sistema herdado possa ser desativado. Esse padrão usa a analogia de uma videira que cresce e se torna uma árvore estabelecida e, eventualmente, supera e substitui sua hospedeira. O padrão foi [apresentado por Martin Fowler](#) como forma de gerenciar riscos ao reescrever sistemas monolíticos. Para ver um exemplo de como aplicar esse padrão, consulte [Modernizar incrementalmente os serviços Web herdados do Microsoft ASP.NET \(ASMX\) usando contêineres e o Amazon API Gateway](#).

sub-rede

Um intervalo de endereços IP na VPC. Cada sub-rede fica alocada em uma única zona de disponibilidade.

controle supervisão e aquisição de dados (SCADA)

Na manufatura, um sistema que usa hardware e software para monitorar ativos físicos e operações de produção.

symmetric encryption (criptografia simétrica)

Um algoritmo de criptografia que usa a mesma chave para criptografar e descriptografar dados.

testes sintéticos

Testar um sistema de forma que simule as interações do usuário para detectar possíveis problemas ou monitorar a performance. Você pode usar o [Amazon CloudWatch Synthetics](#) para criar esses testes.

prompt do sistema

Uma técnica para fornecer contexto, instruções ou orientações a um [LLM](#) a fim de direcionar seu comportamento. Os prompts do sistema ajudam a definir o contexto e a estabelecer regras para interações com os usuários.

T

tags

Pares de valores-chave que atuam como metadados para organizar seus recursos. AWS As tags podem ajudar você a gerenciar, identificar, organizar, pesquisar e filtrar recursos da . Para obter mais informações, consulte [Marcar seus recursos do AWS](#).

variável-alvo

O valor que você está tentando prever no ML supervisionado. Ela também é conhecida como variável de resultado. Por exemplo, em uma configuração de fabricação, a variável-alvo pode ser um defeito do produto.

lista de tarefas

Uma ferramenta usada para monitorar o progresso por meio de um runbook. Uma lista de tarefas contém uma visão geral do runbook e uma lista de tarefas gerais a serem concluídas. Para cada tarefa geral, ela inclui o tempo estimado necessário, o proprietário e o progresso.

ambiente de teste

Veja [ambiente](#).

treinamento

O processo de fornecer dados para que seu modelo de ML aprenda. Os dados de treinamento devem conter a resposta correta. O algoritmo de aprendizado descobre padrões nos dados de treinamento que mapeiam os atributos dos dados de entrada no destino (a resposta que você deseja prever). Ele gera um modelo de ML que captura esses padrões. Você pode usar o modelo de ML para obter previsões de novos dados cujo destino você não conhece.

gateway de trânsito

Um hub de trânsito de rede que você pode usar para interconectar sua rede com VPCs a rede local. Para obter mais informações, consulte [O que é um gateway de trânsito](#) na AWS Transit Gateway documentação.

fluxo de trabalho baseado em troncos

Uma abordagem na qual os desenvolvedores criam e testam recursos localmente em uma ramificação de recursos e, em seguida, mesclam essas alterações na ramificação principal. A ramificação principal é então criada para os ambientes de desenvolvimento, pré-produção e produção, sequencialmente.

Acesso confiável

Conceder permissões a um serviço que você especifica para realizar tarefas em sua organização AWS Organizations e em suas contas em seu nome. O serviço confiável cria um perfil vinculado ao serviço em cada conta, quando esse perfil é necessário, para realizar tarefas de gerenciamento para você. Para obter mais informações, consulte [Usando AWS Organizations com outros AWS serviços](#) na AWS Organizations documentação.

tuning (ajustar)

Alterar aspectos do processo de treinamento para melhorar a precisão do modelo de ML. Por exemplo, você pode treinar o modelo de ML gerando um conjunto de rótulos, adicionando rótulos e repetindo essas etapas várias vezes em configurações diferentes para otimizar o modelo.

equipe de duas pizzas

Uma pequena DevOps equipe que você pode alimentar com duas pizzas. Uma equipe de duas pizzas garante a melhor oportunidade possível de colaboração no desenvolvimento de software.

U

incerteza

Um conceito que se refere a informações imprecisas, incompletas ou desconhecidas que podem minar a confiabilidade dos modelos preditivos de ML. Há dois tipos de incertezas: a incerteza epistêmica é causada por dados limitados e incompletos, enquanto a incerteza aleatória é causada pelo ruído e pela aleatoriedade inerentes aos dados. Para obter mais informações, consulte o guia [Como quantificar a incerteza em sistemas de aprendizado profundo](#).

tarefas indiferenciadas

Também conhecido como trabalho pesado, trabalho necessário para criar e operar um aplicativo, mas que não fornece valor direto ao usuário final nem oferece vantagem competitiva. Exemplos de tarefas indiferenciadas incluem aquisição, manutenção e planejamento de capacidade.

ambientes superiores

Veja [ambiente](#).

V

aspiração

Uma operação de manutenção de banco de dados que envolve limpeza após atualizações incrementais para recuperar armazenamento e melhorar a performance.

controle de versões

Processos e ferramentas que rastreiam mudanças, como alterações no código-fonte em um repositório.

emparelhamento da VPC

Uma conexão entre duas VPCs que permite rotear o tráfego usando endereços IP privados. Para ter mais informações, consulte [O que é emparelhamento de VPC?](#) na documentação da Amazon VPC.

Vulnerabilidade

Uma falha de software ou hardware que compromete a segurança do sistema.

W

cache quente

Um cache de buffer que contém dados atuais e relevantes que são acessados com frequência. A instância do banco de dados pode ler do cache do buffer, o que é mais rápido do que ler da memória principal ou do disco.

dados mornos

Dados acessados raramente. Ao consultar esse tipo de dados, consultas moderadamente lentas geralmente são aceitáveis.

função de janela

Uma função SQL que executa um cálculo em um grupo de linhas que se relacionam de alguma forma com o registro atual. As funções de janela são úteis para processar tarefas, como calcular uma média móvel ou acessar o valor das linhas com base na posição relativa da linha atual.

workload

Uma coleção de códigos e recursos que geram valor empresarial, como uma aplicação voltada para o cliente ou um processo de backend.

workstreams

Grupos funcionais em um projeto de migração que são responsáveis por um conjunto específico de tarefas. Cada workstream é independente, mas oferece suporte aos outros workstreams do projeto. Por exemplo, o workstream de portfólio é responsável por priorizar aplicações, planejar ondas e coletar metadados de migração. O workstream de portfólio entrega esses ativos ao workstream de migração, que então migra os servidores e as aplicações.

WORM

Veja [gravação única e várias leituras](#).

WQF

Veja [AWS Workload Qualification Framework](#).

gravação única e várias leituras (WORM)

Um modelo de armazenamento que grava dados uma única vez e evita que os dados sejam excluídos ou modificados. Os usuários autorizados podem ler os dados quantas vezes forem necessárias, mas não podem alterá-los. Essa infraestrutura de armazenamento de dados é considerada [imutável](#).

Z

exploração de dia zero

Um ataque, normalmente malware, que tira proveito de uma [vulnerabilidade zero-day](#).

vulnerabilidade de dia zero

Uma falha ou vulnerabilidade não mitigada em um sistema de produção. Os agentes de ameaças podem usar esse tipo de vulnerabilidade para atacar o sistema. Os desenvolvedores frequentemente ficam cientes da vulnerabilidade como resultado do ataque.

prompt zero shot

Fornecer a um [LLM](#) instruções para realizar uma tarefa, mas sem exemplos (shots) que possam ajudar a orientá-lo. O LLM deve usar seu conhecimento pré-treinado para lidar com a tarefa. A

eficácia dos prompts zero-shot depende da complexidade da tarefa e da qualidade do prompt.

Veja também [prompts few-shot](#).

aplicação zumbi

Uma aplicação que tem um uso médio de CPU e memória inferior a 5%. Em um projeto de migração, é comum retirar essas aplicações.

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.