Migração de bancos de dados Oracle volumosos para ambientes AWS heterogêneos

# AWS Recomendações

# AWS Recomendações: Migração de bancos de dados Oracle volumosos para ambientes AWS heterogêneos

# Table of Contents

# Migração de bancos de dados Oracle volumosos AWS para ambientes multiplataforma

Incheol Roh, Amazon Web Services ()AWS

Março de 2021 (histórico do documento)

Ao migrar um banco de dados Oracle maior que 100 TB do local para a Amazon Web Services (AWS) Cloud, o tempo de inatividade da migração é um fator sério. Em particular, se o sistema for um sistema de missão crítica, como o planejamento global de recursos corporativos (ERP) ou um sistema de execução de manufatura (MES), você deseja reduzir o tempo de inatividade o máximo possível para a continuidade dos negócios.

Há muitas maneiras de migrar bancos de dados Oracle entre sistemas que têm formatos endian diferentes, conforme mencionado em Estratégias para migrar bancos de dados Oracle para o. AWS Uma dessas abordagens são os espaços de tabela transportáveis (XTTS) multiplataforma Oracle, que você pode usar para reduzir o tempo de inatividade da migração de dias para horas.

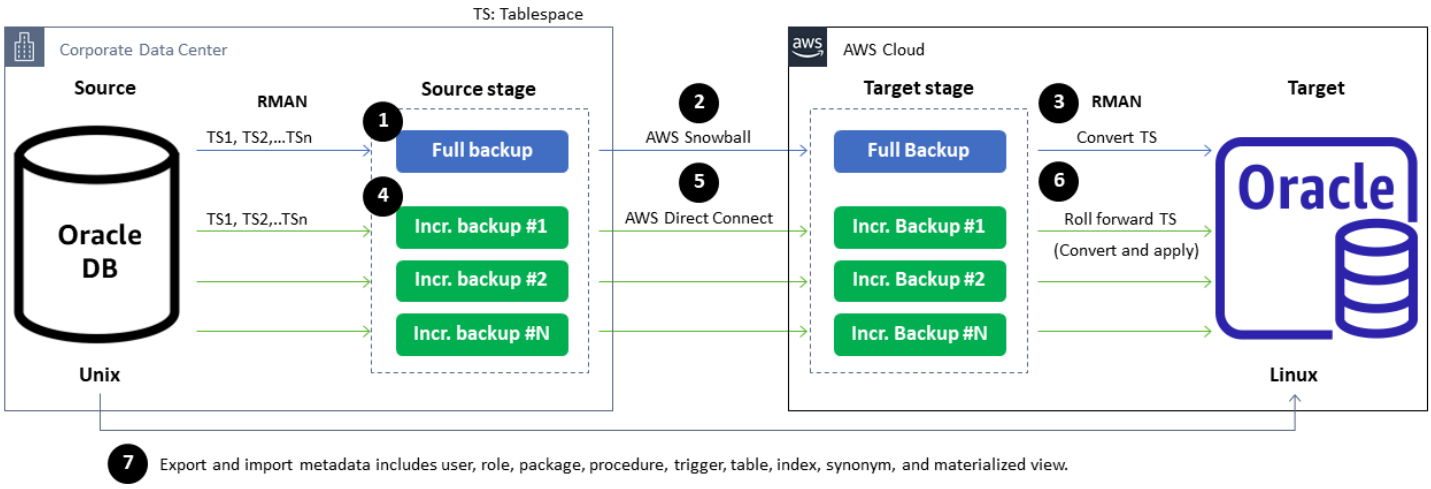
A combinação dos backups incrementais do Oracle XTTS com o Oracle Recovery Manager (RMAN) pode reduzir significativamente a quantidade de tempo de inatividade necessária para mover dados entre plataformas que executam diferentes formatos endian.

Este guia apresenta como usar AWS Snowball EdgeAWS Direct Connect, e o Amazon FSx for Lustre com Oracle XTTS com backups incrementais RMAN. O objetivo dessa abordagem é minimizar o tempo de inatividade da migração em ambientes com conjuntos de dados muito grandes.

# Visão geral do

Esse é o processo conceitual de migração de bancos de dados Oracle para o AWS uso de backups incrementais Oracle XTTS e RMAN com Snowball Edge e FSx for Lustre Direct Connect.

O diagrama a seguir mostra as etapas de migração de alto nível para um banco de dados Oracle em diferentes formatos endian.




1. Faça um backup completo de todos os espaços de tabela.
2. Use o Snowball Edge para mover o backup do estágio de origem para o estágio de destino.
3. Converta os espaços de tabela no banco de dados de destino.
4. Faça backups incrementais.
5. Use Direct Connect para transferir backups incrementais do estágio de origem para o estágio de destino.
6. Avance os backups incrementais, convertendo-os e aplicando-os ao banco de dados de destino.
7. Exporte e importe metadados para todos os espaços de tabela que estão sendo transportados.

Antes da transição, você pode minimizar o tempo de inatividade fazendo o seguinte:

- Exportação e importação de metadados de objetos não baseados em segmentos, incluindo,,, e `USER PACKAGE_SPEC PACKAGE_BODY PROCEDURE FUNCTION`
- Aumento do paralelismo para backup completo e backup incremental
- Convertendo arquivos de dados
- Backups contínuos durante a migração

O documento da Oracle Reduza o Tempo de Inatividade do Tablespace Transportável usando o Backup Incremental de Plataforma Cruzada (2471245.1) explica como usar o Oracle XTTS com backups incrementais do RMAN. O documento também inclui detalhes sobre requisitos e recomendações. O documento não descreve como migrar um banco de dados Oracle de um ambiente local para o Oracle on AWS ou como paralelizar cada etapa da migração para minimizar o tempo de inatividade.

Este guia fornece uma maneira de paralelizar as fases, minimizando o tempo de inatividade da migração em ambientes de sistemas essenciais com tamanhos de dados extremamente grandes.

Após uma fase de configuração inicial, as etapas de alto nível para usar o Oracle XTTS com backups incrementais do RMAN incluem as seguintes fases.

## Fase 1 — Fase de preparação

A fase de preparação consiste nas seguintes etapas:

1. Um backup inicial completo (nível=0) dos espaços de tabela é levado do banco de dados de origem até o estágio de origem, que é o armazenamento NAS.
2. As cópias de backup são transferidas usando o Snowball Edge para o estágio de destino, que é o FSx for Lustre integrado ao Amazon Simple Storage Service (Amazon S3).
3. Os espaços de tabela de backup são restaurados e convertidos no banco de dados de destino com o formato little-endian.

As etapas dessa fase são executadas somente uma vez durante a migração. Os dados que estão sendo transportados estão totalmente acessíveis no banco de dados de origem durante essa fase.

## Fase 2 — Roll-forward fase

A fase de avanço consiste nas seguintes etapas:

1. Um backup incremental é levado do banco de dados de origem para o estágio de origem.
2. As cópias de backup incrementais são transferidas para o estágio de Direct Connect destino.
3. As cópias de backup incrementais são convertidas no banco de dados de destino com o formato little-endian. As cópias são então aplicadas ao banco de dados de destino inicial, o que é chamado de etapa de roll-forward.

Você pode executar essa fase várias vezes. Cada backup incremental sucessivo deve levar menos tempo e tornará as cópias do arquivo de dados de destino mais atualizadas com o banco de dados de origem. Como na fase 1, os dados de origem que estão sendo transportados estão totalmente acessíveis durante essa fase.

## Fase 3 — Fase de transporte

A terceira fase inclui as seguintes etapas:

1. Os espaços de tabela que estão sendo transportados são alterados para somente leitura.
2. Um backup incremental final é obtido do banco de dados de origem.
3. Os metadados são exportados.
4. Os backups são transferidos e aplicados ao destino.
5. Os metadados do objeto são importados.

Nesse ponto, o número de alteração do sistema (SCN) do banco de dados de destino é consistente com o do banco de dados de origem.

Os metadados dos espaços de tabela transportáveis são exportados do banco de dados de origem e importados para o banco de dados de destino. Os metadados incluem informações sobre o usuário, a função, o pacote, o procedimento, a função, a tabela e o índice.

Finalmente, os espaços de tabela são criados read/write para acesso total ao banco de dados de destino a partir do aplicativo.

Essa fase é seguida por uma fase de validação.

# Fases de migração

Este guia aborda a migração da instância única Oracle e do Oracle RAC como banco de dados de origem e destino. Se a origem e o destino forem o Oracle RAC, você poderá maximizar o paralelismo para cada etapa da migração.

## Fase 0 — Fase de configuração inicial

1. Instale o banco de dados Oracle 12.1.0.2 ou posterior na instância de destino do Amazon Elastic Compute Cloud (Amazon EC2).
2. Verifique o banco de dados de destino.

   Você deve verificar se a instância do banco de dados de destino tem o mesmo fuso horário e conjunto de caracteres do banco de dados de origem. Caso contrário, essa abordagem de migração falhará

   Para verificar se a versão do fuso horário na origem e no destino é a mesma, execute o comando a seguir.

   ```
   SQL> select * from v$timezone_file;
   FILENAME            VERSION    CON_ID
   ----------------------   -------------   ----------
   timezlrg_14.dat        14               0
   ```

   Para verificar se o conjunto de caracteres do banco de dados e o conjunto de caracteres nacional são iguais na origem e no destino, execute o comando a seguir.

   ```
   SQL> select * from nls_database_parameters
            where parameter in ('NLS_CHARACTERSET','NLS_NCHAR_CHARACTERSET');
   PARAMETER                                VALUE
   ----------------------------------------   --------------------
   NLS_NCHAR_CHARACTERSET         UTF8
   NLS_CHARACTERSET                    AL32UTF8
   ```
3. Configure o utilitário Oracle XTTS.

No sistema de origem, baixe e extraia o arquivo de script VER4.ziprman_xttconvert_ do documento Oracle 2471245.1. Modifique os parâmetros no `xtt.properties` arquivo com a configuração específica. Em seguida, `xtt.properties` copie `xttdriver.pl` os scripts para o sistema de destino.

# Fase 1 — Fase de preparação (o banco de dados de origem permanece on-line)

Durante essa fase, os arquivos de dados do espaço de tabela a ser transportado são copiados no sistema de origem. As cópias de backup são transportadas para o sistema de destino e restauradas com a execução do `xttdriver.pl` script.

## Etapa 1: Fazer backups completos (nível=0) do tablespace no sistema de origem

Para reduzir o tempo de duração do backup, `xtt.properties` copie os `xttdriver.pl` arquivos e para vários diretórios. Em cada `xtt.properties` arquivo em cada diretório, insira o nome do espaço de tabela no parâmetro. `tablespaces` Em seguida, em cada diretório, execute `xttdriver.pl` com a `--backup` opção. Esse comando transporta todos os espaços de tabela`SYSTEM`, exceto,,, e `SYSAUX` `UNDOTEMP`, que foram criados durante a instalação do banco de dados de destino.

Por exemplo, cada `xtt01~xtt04` diretório tem todos os `xtt` scripts (`xtt.propertiesexttdriver.pl`) com valores diferentes para o `tablespaces=` parâmetro no `xtt.properties` arquivo. Ao editar o `tablespaces` parâmetro em cada `xtt.properties` arquivo, considere dividir os grupos de tablespaces para que o tamanho total dos arquivos de dados em cada grupo de tablespace seja semelhante.

Neste guia, o exemplo supõe que há quatro grupos de espaços de tabela. Se o banco de dados de origem for uma instância única Oracle, você criará todos os `xtt01~04` diretórios. Se o banco de dados de origem for o Oracle RAC, você poderá criar cada `xtt` diretório em cada servidor Oracle RAC.

```
[oracle@erp expimp]$ ls
out  xtt01  xtt02  xtt03    xtt04
[oracle@erp out]$ ls
```

```
out01 out02 out03 out04
```

A tabela a seguir mostra exemplos do `tablespaces=` parâmetro nos `xtt.properties` arquivos.

| | |
|---|---|
| xtt01 | tablespaces=APPS_TS_TX_DATA |
| xtt02 | tablespaces=APPS_TS_TX_IDX |
| xtt03 | tablespaces=APPS_TS_SUMMARY |
| xtt04 | tablespaces=APPS_CALCLIP, APPS_OMO, APPS_TS_ARCHIVE, APPS_TS_DISCO, APPS_TS_DISCO_OLAP , APPS_TS_INTERFACE, APPS_TS_M EDIA, APPS_TS_NOLOGGING, APPS_TS_QUEUES, APPS_TS_SEED… |

Para evitar que qualquer cópia de backup de arquivos de dados existente seja excluída enquanto `xttdriver.pl` estiver sendo executada em paralelo, comente a seguinte linha em`xttdriver.pl`.

```
   if (@present)
   {
      ## Try deleting any existing copied datafiles
##Comment out it for executing rman command in parallel by AWS
##        system("\\rm -f $dfcopydir/*.tf");
      sleep 5;
   }
```

Use `xttdriver.pl` para fazer o backup RMAN do sistema de origem.

Se o sistema de origem for o Oracle RAC, ele poderá ser executado em cada instância do Oracle RAC simultaneamente.

A saída de `xttdriver.pl` é armazenada na variável de ambiente `TMPDIR`. Execute cada `xttdriver.pl` comando com a `--backup` opção e use a `--debug` opção para ativar o modo de depuração.

```
cd /u01/oracle/expimp/xtt<nn>
export TMPDIR=/u01/oracle/expimp/out/out<nn>
```

```
$ORACLE_HOME/perl/bin/perl xttdriver.pl --backup --debug 3
```

Neste exemplo, <nn>representa os grupos de espaços de tabela. Se houver quatro grupos de espaços de tabela, <nn>torna-se, `0102`, e. `03 04`

## Etapa 2: Transferir cópias de backup completas para o sistema de destino

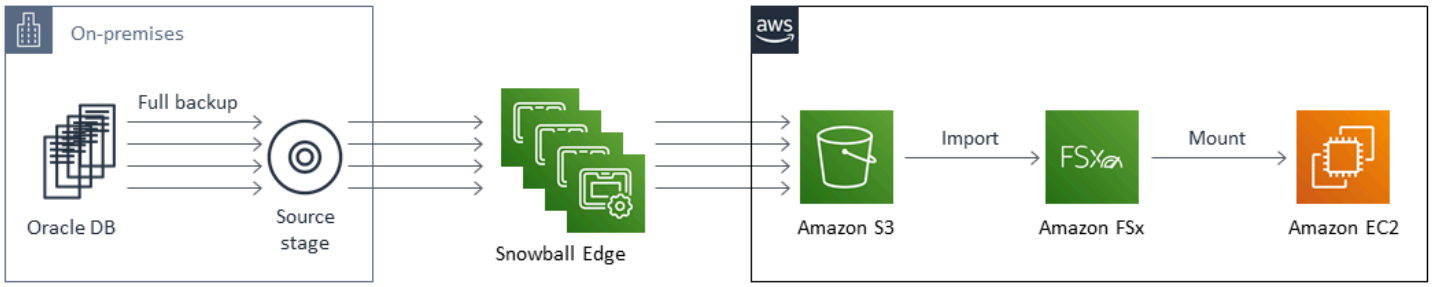
Use uma das duas opções a seguir para transferir suas cópias de backup para o sistema de destino.

### Opção 1 — Usando AWS Snowball Edge

Caminho: Banco de dados de origem -> estágio de origem -> AWS Snowball Edge -> Amazon S3 -> FSx for Lustre

> **Note**
>
> AWS Snowball Edge não está mais disponível para novos clientes. Novos clientes devem explorar AWS DataSynctransferências on-line, o Terminal de Transferência de AWS Dados para transferências físicas seguras ou AWS Partner soluções. Para computação de ponta, explore AWS Outposts.

O diagrama a seguir mostra a transferência de um backup completo usando o Snowball Edge.




O Snowball Edge é um dispositivo físico robusto de computação e armazenamento que você pode usar para mover dados em grande escala. AWS O Snowball Edge ajuda a superar os desafios que você pode encontrar com transferências de dados em grande escala, incluindo longos tempos de transferência, falta de largura de banda utilizável e questões de segurança.

Todas as cópias de backup do arquivo de dados são transferidas para o Amazon S3 pelo Snowball Edge. Se o tamanho do sistema de origem for superior a 100 TB, você deverá usar vários dispositivos Snowball Edge para reduzir a duração da transferência.

O Amazon FSx para Lustre está profundamente integrado ao Amazon S3. Você pode criar um sistema de arquivos FSx for Lustre vinculado ao seu bucket do S3 em minutos. Quando vinculados ao repositório de dados do Amazon S3, os sistemas de arquivos FSx for Lustre apresentam de forma transparente os objetos do Amazon S3 como arquivos. No ambiente real, o desempenho do FSx for Lustre depende da capacidade de armazenamento, do tamanho de cada arquivo e da forma como os arquivos são distribuídos no sistema de arquivos. Quando o FSx for Lustre é usado como armazenamento em estágio de destino, você não precisa restaurar cópias de backup do tablespace no Amazon Elastic Block Store (Amazon EBS) ou no Amazon Elastic File System a partir do Amazon S3.

1. Importe o bucket do S3 para o FSx for Lustre.

   Use o Snowball Edge para transferir e armazenar a área do palco de origem (por exemplo,`src_backups`) em um bucket do S3 (por exemplo,). `s3-src-backups`

   Crie o sistema de arquivos FSx for Lustre (por `dest_backups` exemplo,) como a área de estágio de destino para as cópias de backup dos arquivos de dados.

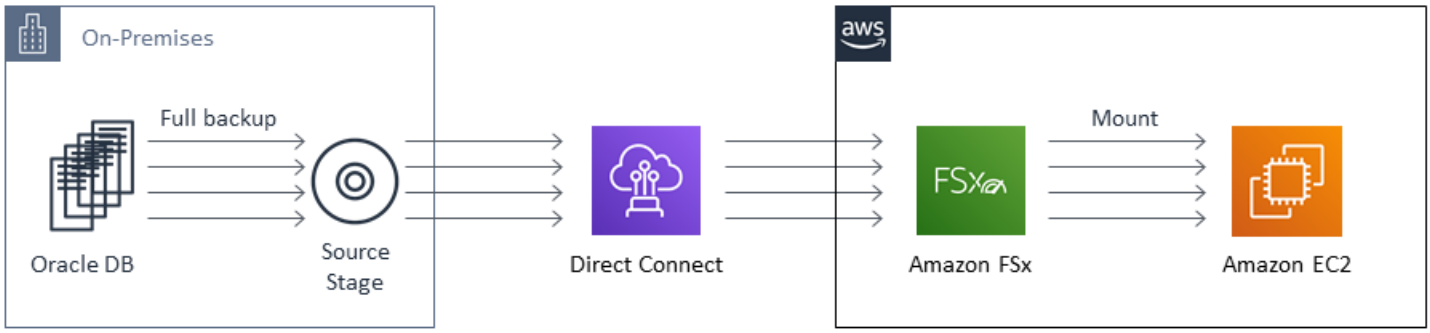
   Instale o cliente Lustre de código aberto no sistema de destino (Amazon EC2). Para obter mais informações, consulte o Guia do usuário do Amazon FSx para Lustre.

   Depois que o cliente Lustre for instalado, monte o sistema de arquivos FSx for Lustre no sistema de destino (Amazon EC2).
2. Transferir `res.txt` da `$TMPDIR` origem para `$TMPDIR` o destino.

## Opção 2 — Usando AWS Direct Connect

Caminho: Banco de dados de origem -> estágio de origem -> AWS Direct Connect -> FSx for Lustre

O diagrama a seguir mostra a transferência de um backup completo usando AWS Direct Connect.




Direct Connect oferece a capacidade de criar conexões de rede privadas entre seu data center, escritório ou ambiente de colocation e. AWS Essas conexões de rede privada reduzem os custos da rede, aumentam a taxa de transferência e oferecem uma experiência consistente.

Você pode transferir diretamente cópias de backup de arquivos de dados do sistema de origem para o sistema de destino (Amazon EC2) onde o sistema de arquivos Amazon FSx for Lustre está montado. Essa opção é mais rápida do que a opção Snowball Edge se você puder acomodar a alta largura de banda do. Direct Connect

Depois que as cópias de backup dos arquivos de dados forem transferidas pelo Snowball Edge ou Direct Connect, você poderá vê-las no sistema de arquivos FSx for Lustre na área do estágio de destino.

## Etapa 3: restaurar cópias de backup completas e converter arquivos de dados

Restaure cópias de backup completas e converta arquivos de dados no formato endian do sistema de destino. Para reduzir a duração da restauração e da conversão, execute o `xttdriver.pl` comando com a `--restore` opção para cada grupo de espaços de tabela.

```
cd /u01/oracle/expimp/xtt<nn>
export TMPDIR=/u01/oracle/expimp/out/out<nn>
$ORACLE_HOME/perl/bin/perl xttdriver.pl --restore --debug 3
```

Após a conclusão da etapa, os arquivos de dados são colocados no dest_datafile_location definido no arquivo no sistema de destino. `xtt.properties`

# Fase 2 — Roll-forward fase (o banco de dados de origem permanece on-line)

Você pode repetir a fase de roll-forward quantas vezes for necessário para capturar o arquivo de dados de destino até o banco de dados de origem.

A fase de avanço consiste nas seguintes etapas.

1. Crie um backup incremental do banco de dados de origem.
2. Transfira o backup para o sistema de destino.
3. Converta o backup para o formato endian do sistema de destino.
4. Aplique o backup às cópias convertidas do arquivo de dados de destino para transferi-las para frente.

Cada backup incremental sucessivo deve levar menos tempo e tornará as cópias do arquivo de dados de destino mais atualizadas com o banco de dados de origem.

## Etapa 1: Faça backups incrementais em paralelo

Faça backups incrementais dos grupos de espaços de tabela que estão sendo transportados no sistema de origem em paralelo.

Essa etapa cria backups incrementais para todos os tablespaces= listados no arquivo. `xtt.properties`

Se você puder ativar o recurso BLOCK CHANGE TRACKING no sistema de origem, poderá reduzir consideravelmente o tempo do backup incremental.

O comando a seguir reconhece automaticamente o próximo SCN após o backup completo.

```
cd /u01/oracle/expimp/xtt<nn>
export TMPDIR=/u01/oracle/expimp/out/out<nn>
$ORACLE_HOME/perl/bin/perl xttdriver.pl --backup --debug 3
```

## Etapa 2: Transferir os backups incrementais e o arquivo res.txt para o sistema de destino

Se você tiver largura de banda suficiente, poderá reduzir a duração da transferência usando. Direct Connect

Transfira os backups incrementais entre `src_scratch_location` e. `dest_scratch_location` Transfira o `res.txt` arquivo entre `$TMPDIR` o sistema de origem e `$TMPDIR` o sistema de destino.

Se você fizer vários backups incrementais, o `res.txt` arquivo deverá ser copiado após o último backup incremental antes de poder ser aplicado no sistema de destino.

```
[source]$ scp $TMPDIR/res.txt oracle@[dest]:/u01/oracle/expimp/out/out<nn>
[source]$ scp <src_scratch_location>/* oracle@[dest]:<dest_scratch_location>
```

## Etapa 3: converter e aplicar backups incrementais

Converta backups incrementais no formato endian do sistema de destino e aplique backups incrementais às cópias do arquivo de dados de destino no sistema de destino.

Neste guia, suponha que os grupos de espaços de tabela sejam quatro. Execute cada `xttdriver.pl` comando com a `--restore` opção para cada grupo de espaços de tabela.

Nesta etapa, o utilitário Oracle XTTS faz com que o banco de dados de destino seja reiniciado. Para executar o comando em paralelo, você deve usar a seguinte personalização no script Perl,. `xttdriver.pl`

1. Comente as seguintes linhas:
   - Linha 4867: `my $outputstart = `sqlplus -L -s \"/ as sysdba\" \@xttstartupnomount.sql`;`
   - Linha 4868: `checkError ("Error in executing xttstartupnomount.sql", $outputstart);`
   - Linha 4992: `my $outputstart = `sqlplus -L -s \"/ as sysdba\" \@xttdbopen.sql`;\`
2. Coloque o banco de dados de destino em `NOMOUNT` status.

```
sqlplus / as sysdba @xttstartupnomount.sql
```

3. Execute uma expansão dos backups incrementais em paralelo.

```
cd /u01/oracle/expimp/xtt<nn>
export TMPDIR=/u01/oracle/expimp/out/out<nn>
$ORACLE_HOME/perl/bin/perl xttdriver.pl --restore --debug 3
```

4. Depois de avançar com todos os backups incrementais, defina o `OPEN` status do banco de dados de destino.

```
sqlplus / as sysdba @xttdbopen.sql
```

Nesse ponto, você pode repetir a fase 2 até que os SCNs nos bancos de dados de origem e destino se aproximem o suficiente. Você deve decidir se vai para a fase 3 ou se repete a fase 2, dependendo do tempo de inatividade desejado.

Para reduzir o tempo de inatividade durante a fase 3 (a fase de transporte), você pode exportar e importar os metadados de objetos não baseados em segmentos, como,, e `USER` `PACKAGE` `PROCEDUREFUNCTION`, antes de definir o banco de dados de origem como. `READ ONLY`

Se o número de objetos do banco de dados no banco de dados de origem for extremamente grande (centenas de milhares), a exportação e importação dos metadados levará várias horas. Nesse caso, considere exportar metadados.

A exportação de objetos não baseados em segmentos é executada read/write no sistema de origem. Em seguida, você deve importar os objetos para o sistema de destino antes que o sistema de origem seja definido como`READ ONLY`. Neste momento, você deve manter os objetos de origem do banco de dados`PACKAGE`, como, e`PROCEDURE`, `FUNCTION` inalterados.

Esse é um exemplo do arquivo de parâmetros de despejo usado para exportar metadados de objetos não baseados em segmentos, incluindo,,`USER`, e. `PACKAGE_SPEC` `PACKAGE_BODY` `PROCEDURE` `FUNCTION`

```
directory=dmpdir
dumpfile=xttsmsc%U.dmp
full=y
filesize=1048576000
```

```
logfile=expmsc.log
metrics=y
exclude=TABLE,INDEX,CONSTRAINT,COMMENT,
MATERIALIZED_VIEW,MATERIALIZED_VIEW_LOG,SCHEMA_CALLOUT
```

Use o comando a seguir para exportar metadados.

```
SQL> create directory dmpdir as <location>;
expdp system/<system password> parfile=<parameter file>
```

Esse é um exemplo do arquivo de parâmetros de despejo para importar metadados de objetos não segmentados.

```
directory=dmpdir
dumpfile=xttsmsc%U.dmp
full=y
logfile=impmsc1.log
EXCLUDE=TABLESPACE, PROCOBJ, RLS_CONTEXT, RLS_GROUP, RLS_POLICY, TABLESPACE_QUOTA
metrics=y
remap_tablespace=
APPS_TS_ARCHIVE:SYSTEM,
APPS_TS_INTERFACE:SYSTEM,
APPS_TS_SEED:SYSTEM,
APPS_TS_SUMMARY:SYSTEM,
…..
```

No momento, não há espaços de tabela`SYSTEM`, exceto, `SYSAUXUNDO`, e `TEMP` no sistema de destino. Portanto, você deve remapear todos os objetos a serem importados para o espaço de `SYSTEM` tabela.

Use o comando a seguir para importar metadados.

```
SQL> create directory dmpdir as <location>;
impdp system/<system password> parfile=<parameter file>
```

Agora você pode ver os objetos criados no banco de dados de destino.

```
SQL> select object_type, count(*) from dba_objects group by object_type order by
 count(*) desc;
OBJECT_TYPE                 COUNT(*)
```

```
------------------------------ ----------
SYNONYM                         89327
PACKAGE                          55670
PACKAGE BODY                   54447
VIEW                                41378
JAVA CLASS                       31978
SEQUENCE                        12766
…..
```

Não há espaços de tabelaSYSTEM, exceto,SYSAUX, e. UNDO TEMP O USER objeto é criado com USERS o tablespace padrão. Durante a fase 3, os espaços de tabela transportáveis serão criados e, em seguida, os objetos USER serão alterados com os espaços de tabela padrão originais.

# Fase 3 — Fase de transporte (o banco de dados de origem é somente para leitura)

Durante essa fase, o sistema de origem fica somente para leitura. Os arquivos de dados no sistema de destino são tornados consistentes com o sistema de origem aplicando um backup incremental final. Em seguida, exporte os metadados do objeto do sistema de origem e importe-os para o sistema de destino.

## Etapa 1: Tornar os espaços de tabela no banco de dados de origem somente para leitura

Como SYSDBA, crie todos os espaços de tabela que estão sendo transferidos READ ONLY no sistema de origem.

Para reduzir o tempo de inatividade, você pode executar as duas etapas a seguir simultaneamente.

## Etapa 2: criar o backup incremental final

No sistema de origem, crie o backup incremental final dos espaços de tabela que estão sendo transferidos.

```
cd /u01/oracle/expimp/xtt<nn>
export TMPDIR=/u01/oracle/expimp/out/out<nn>
$ORACLE_HOME/perl/bin/perl xttdriver.pl --backup --debug 3
```

Essa etapa retorna um erro, como "ORA-20001: TABLESPACE (S) IS READONLY"; o erro é esperado e você pode ignorá-lo com segurança.

## Etapa 3: exportar os metadados

Exporte os metadados dos espaços de tabela transportáveis do banco de dados de origem.

Esse é um exemplo do arquivo de parâmetros para exportar os metadados dos espaços de tabela transportáveis.

```
directory=dmpdir
metrics=y
dumpfile=xttsmeta%U.dmp
filesize=1048576000
logfile=expxtts.log
transport_tablespaces=
APPS_TS_ARCHIVE,
APPS_TS_INTERFACE,
APPS_TS_MEDIA,
APPS_TS_NOLOGGING,
….
exclude=table_statistics,index_statistics
```

Além disso, se o sistema de origem tiver muitas tabelas e índices, você poderá economizar tempo durante a exportação excluindo suas estatísticas. Depois de importar o espaço de tabela transportável, importe as estatísticas para o sistema de destino.

Antes de executar`expdp`, crie um diretório de banco de dados no qual os arquivos de despejo sejam armazenados no sistema de origem.

```
SQL> create directory dmpdir as <location>;
expdp system/<system password> parfile=<parameter file>
```

As duas etapas a seguir são as etapas finais para um espaço de tabela transportável entre plataformas com backups incrementais do RMAN. Essas etapas devem ser executadas sequencialmente.

## Etapa 4: Transferir os arquivos e aplicar o backup incremental final

Transfira o backup incremental final e exporte os arquivos de despejo para o sistema de destino, converta e aplique o backup incremental final.

Use Direct Connect para transferir as cópias de backup incrementais finais e o arquivo res.txt para o destino. Você pode usar a conectividade VPN, mas o uso Direct Connect reduzirá significativamente o tempo de inatividade se houver largura de banda suficiente.

Para restaurar o backup incremental final, no sistema de destino, execute o comando a seguir, usando a `--restore` opção, para cada grupo de tablespace.

```
cd /u01/oracle/expimp/xtt<nn>
export TMPDIR=/u01/oracle/expimp/out/out<nn>
$ORACLE_HOME/perl/bin/perl xttdriver.pl --restore --debug 3
```

## Etapa 5: importar os metadados do objeto

Importe os metadados do objeto para o sistema de destino usando o Oracle Data Pump. Execute o comando a seguir para obter a lista de arquivos de dados para o `transport_datafiles=` parâmetro no sistema de destino.

```
cd /u01/oracle/expimp/xtt<nn>
export TMPDIR=/u01/oracle/expimp/out/out<nn>
$ORACLE_HOME/perl/bin/perl xttdriver.pl -e
```

Sempre que você executa o comando anterior, você obtém o `xttplugin.txt` arquivo, que tem o `transport_datafiles=` parâmetro. Mescle `transport_datafiles=` em uma única linha todos os `xttplugin.txt` arquivos e coloque as listas de arquivos de dados no `transport_datafiles` argumento do arquivo de parâmetros para os metadados de importação.

O trecho de código a seguir mostra o arquivo de parâmetros para importar o espaço de tabela transportável no sistema de destino.

```
directory=dmpdir
metrics=y
dumpfile=xttsmeta%U.dmp
logfile=impxtts.log
exclude=TYPE
transport_datafiles= '+EBSDATA/APPS_TS_TX_DATA_2.dbf','+EBSDATA/
APPS_TS_TX_DATA_11.dbf','+EBSDATA/APPS_TS_TX_DATA_22.dbf','+EBSDATA/
APPS_TS_TX_DATA_183.dbf','+EBSDATA/APPS_TS_TX_DATA_204.dbf','+EBSDATA/
APPS_TS_TX_DATA_219.dbf','+EBSDATA/APPS_TS_TX_DATA_227.dbf'…..
```

Antes de executar`impdp`, crie um diretório de banco de dados apontando para o local dos arquivos de despejo de exportação.

```
SQL> create directory dmpdir as <location>;
impdp system/<system password> parfile=<parameter file>
```

# Fase 4 — Validar os dados transportados

## Etapa 1: Verificar se há corrupção nos espaços de tabela

Nesta etapa, os espaços de tabela transportados são lidos somente no banco de dados de destino. Você pode validar se os espaços de tabela são válidos ou não executando o comando RMAN da seguinte forma.

```
sqlplus / as sysdba;
set feedback off
set pagesize 0
set heading off
spool tbs_val.sql;
select 'validate tablespace '||tablespace_name||' check
 logical;' from dba_tablespaces where tablespace_name not in
 ('SYSTEM','SYSAUX','TEMP','USERS','UNDOTBS1','UNDOTBS2','UNDOTBS3','UNDOTBS3','UNDOTBS4');
spool off;
exit;

--Remove the first and last line in the spool file, tbs_val.sql
rman target /
RMAN> @tbs_val.sql
```

Além disso, você pode verificar a contagem dos objetos, como os objetos de tabela, índice, sinônimo, visualização e pacote.

## Etapa 2. Crie todos os espaços de tabela transportados no banco de dados de destino read/write

```
sqlplus / as sysdba;
set feedback off
set pagesize 0
set heading off
```

```
spool tbs_rw.sql
select 'alter tablespace '||tablespace_name||' read write;' from dba_tablespaces where
 status='READ ONLY';
spool off;
exit;
--Remove the first and last line in the spool file, tbs_rw.sql
sqlplus / as sysdba;
SQL> @tbs_rw.sql
```

# Conclusão

Neste guia, você aprendeu como minimizar o tempo de inatividade usando espaços de tabela transportáveis entre plataformas da Oracle e backups incrementais do RMAN com AWS Snowball Edge,, e o Amazon for Lustre. AWS Direct Connect FSx

Ao migrar um banco de dados Oracle do local para AWS, considere as seguintes variáveis:

- A largura de banda da rede exigida pelo Direct Connect
- A capacidade do FSx For Lustre
- O tamanho dos metadados do banco de dados Oracle
- O tamanho do backup incremental

Usando o método recomendado neste guia, ao migrar de um banco de dados Oracle de 100 TB executado em um sistema Unix para AWS, você pode reduzir o tempo de inatividade para cerca de 10 horas.

# Histórico do documento

A tabela a seguir descreve mudanças significativas neste guia. Se desejar receber notificações sobre futuras atualizações, inscreva-se em um feed RSS.

| Alteração | Descrição | Data |
| --- | --- | --- |
| – | Menções alteradas de heterogêneo na Introdução e Visão Geral. | 14 de julho de 2021 |
| – | Publicação inicial | 2 de março de 2021 |

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.