将庞大的 Oracle 数据库迁移到 AWS 异构环境中

AWS 规范性指导

AWS 规范性指导: 将庞大的 Oracle 数据库迁移到 AWS 异构环境中

Table of Contents

将庞大的 Oracle 数据库迁移到 AWS 适用于跨平台环境

Incheol Roh，Amazon Web Services ()AWS

2021 年 3 月（文档历史记录）

将大于 100 TB 的 Oracle 数据库从本地迁移到 Amazon Web Services (AWS) 云时，迁移停机是一个严重的因素。特别是，如果系统是任务关键型系统，例如全球企业资源规划 (ERP) 或制造执行系统 (MES)，则您希望尽可能减少停机时间以实现业务连续性。

有许多方法可以在具有不同字节序格式的系统之间迁移 Oracle 数据库，如将 Oracle 数据库迁移到的策略中所述。 AWS其中一种方法是 Oracle 跨平台可传输表空间 (XTTS)，您可以使用它来将迁移停机时间从几天缩短到几小时。

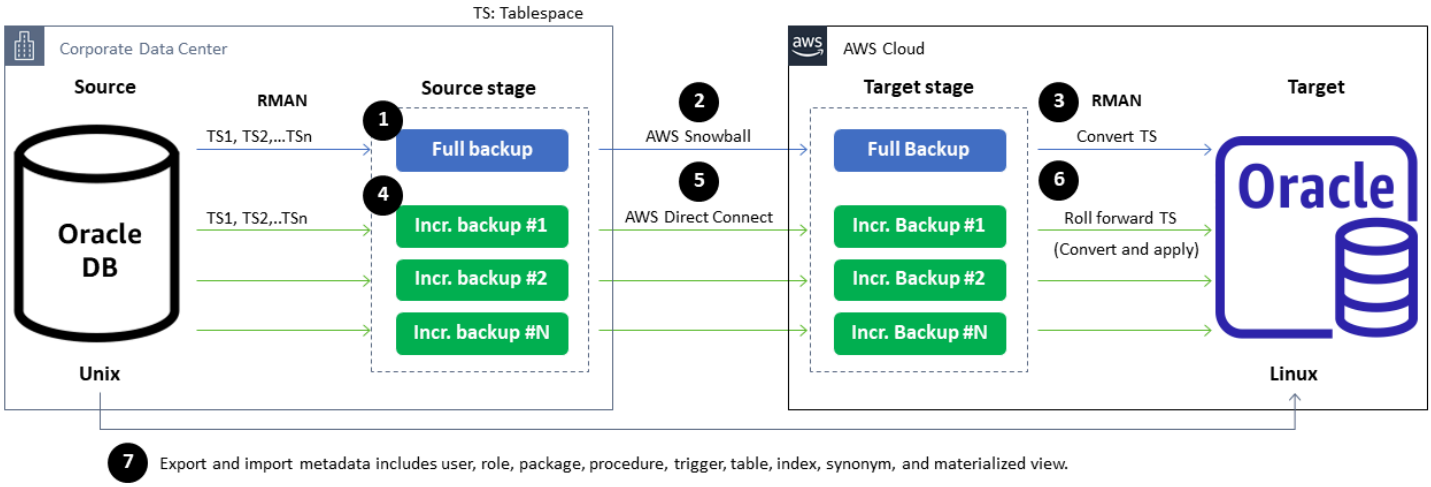
将 Oracle XTTS 与 Oracle Recovery Manager (RMAN) 增量备份相结合，可以显著减少在运行不同字节序格式的平台之间移动数据所需的停机时间。

本指南介绍了如何使用带有 RMAN 增量AWS Snowball Edge备份AWS Direct Connect的 Oracle XTTS 和 Amazon FSx for Lust re。这种方法的目标是在数据集非常大的环境中最大限度地减少迁移停机时间。

概述

这是将 Oracle 数据库迁移到 AWS 使用 Oracle XTTS 和 RMAN 增量备份和 Snowball Edge 和 FS Direct Connect x for Lustre 的概念过程。

下图显示了跨不同字节序格式的 Oracle 数据库的高级迁移步骤。



1. 对所有表空间进行完整备份。
2. 使用 Snowball Edge 将备份从源阶段移动到目标阶段。
3. 将表空间转换为目标数据库。
4. 进行增量备份。
5. 用于 Direct Connect 将增量备份从源阶段传输到目标阶段。
6. 向前滚动增量备份，将其转换并应用于目标数据库。
7. 导出和导入所有正在传输的表空间的元数据。

在转换之前，您可以通过执行以下操作来最大限度地减少停机时间：

- 导出和导入非基于分段的对象的元数据，包括、USER、PACKAGE_SPEC、PACKAGE_BODY和 PROCEDURE FUNCTION
- 提高完整备份和增量备份的并行度
- 转换数据文件
- 迁移期间向前滚动备份

Oracle 文档《使用跨平台增量备份减少可传输表空间停机时间》(2471245.1) 解释了如何使用 Oracle XTTS 和 RMAN 增量备份。该文件还包括有关要求和建议的详细信息。本文档没有描述如何将 Oracle 数据库从本地环境迁移到 Oracle，也没有描述如何并行执行每个迁移步骤以最大限度地减少停机时间。 AWS

本指南提供了一种并行化各个阶段的方法，最大限度地减少了数据量极大的任务关键型系统环境中的迁移停机时间。

在初始设置阶段之后，使用 Oracle XTTS 和 RMAN 增量备份的高级步骤包括以下阶段。

第 1 阶段 — 准备阶段

准备阶段包括以下步骤：

1. 将源数据库上表空间的初始完整备份（级别 =0）带到源阶段（即 NAS 存储）。
2. 使用 Snowball Edge 将备份副本传输到目标阶段，即与亚马逊简单存储服务 (Amazon S3) Simple Service 集成的FSx for Lustre。
3. Backup 表空间被还原并转换为小端格式的目标数据库。

此阶段的步骤在迁移期间仅运行一次。在此阶段，正在传输的数据可以在源数据库中完全访问。

第 2 阶段 — Roll-forward 阶段

向前滚动阶段包括以下步骤：

1. 从源数据库到源阶段进行增量备份。
2. 增量备份副本将传输到目标阶段 Direct Connect。
3. 增量备份副本以小端格式转换为目标数据库。然后将这些副本应用于初始目标数据库，这称为向前滚步骤。

您可以多次运行此阶段。每次连续的增量备份都应花费更少的时间，并且会使目标数据文件副本与源数据库保持一致。与第 1 阶段一样，在此阶段可以完全访问正在传输的源数据。

第 3 阶段 — 传输阶段

第三阶段包括以下步骤：

1. 正在传输的表空间更改为只读。
2. 最后一次增量备份取自源数据库。
3. 元数据已导出。
4. 备份已传输并应用到目标。
5. 已导入对象元数据。

此时，目标数据库的系统更改号 (SCN) 与源数据库的系统更改号 (SCN) 一致。

可传输表空间的元数据从源数据库导出并导入到目标数据库中。元数据包括用户、角色、包、过程、函数、表和索引的信息。

最后，表空间 read/write 用于从应用程序对目标数据库进行完全访问。

此阶段之后是验证阶段。

迁移阶段

本指南介绍如何同时迁移 Oracle 单实例和 Oracle RAC 作为源数据库和目标数据库。如果源和目标是 Oracle RAC，则可以最大限度地提高每个迁移步骤的并行度。

第 0 阶段 — 初始设置阶段

1. 在目标亚马逊弹性计算云 (Amazon EC2) 实例上安装 Oracle 数据库 12.1.0.2 或更高版本。
2. 验证目标数据库。

   您必须验证目标数据库实例是否与源数据库具有相同的时区和字符集。否则，这种迁移方法将失败

   要检查源端和目标端的时区版本是否相同，请运行以下命令。

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

   要检查源和目标中的数据库字符集和国家字符集是否相同，请运行以下命令。

```
SQL> select * from nls_database_parameters
         where parameter in ('NLS_CHARACTERSET','NLS_NCHAR_CHARACTERSET');
PARAMETER                                VALUE
-----------------------------------------   --------------------
NLS_NCHAR_CHARACTERSET         UTF8
NLS_CHARACTERSET                    AL32UTF8
```

3. 设置 Oracle XTTS 实用程序。

   在源系统上，从 Oracle 文档 2471245.1 下载并提取脚本文件 rman_xttconvert VER4.zip _。使用特定配置修改`xtt.properties`文件中的参数。`xtt.properties`然后将`xttdriver.pl`脚本复制到目标系统。

第 1 阶段 — 准备阶段（源数据库保持在线）

在此阶段，将在源系统上备份要传输的表空间的数据文件。Backup 副本被传输到目标系统并通过运行xttdriver.pl脚本进行恢复。

步骤 1：对源系统进行完整（级别=0）表空间备份

要缩短备份持续时间，请将xtt.properties和xttdriver.pl文件复制到多个目录。在每个目录下的每个xtt.properties文件中，在tablespaces参数中输入表空间的名称。然后，在每个目录下，xttdriver.pl使用--backup选项运行。此命令传输除安装目标数据库期间创建的SYSTEMSYSAUX、UNDOTEMP、和之外的所有表空间。

例如，每个xtt01~xtt04目录都有所有xtt脚本（xtt.properties和xttdriver.pl），xtt.properties文件中tablespaces=参数的值各不相同。编辑每个xtt.properties文件中的tablespaces参数时，请考虑划分表空间组，使每个表空间组中数据文件的总大小相似。

在本指南中，该示例假设有四个表空间组。如果源数据库是 Oracle 单实例，则创建所有xtt01~04目录。如果源数据库是 Oracle RAC，则可以在每台 Oracle RAC 服务器中创建每个xtt目录。

```
[oracle@erp expimp]$ ls
out  xtt01  xtt02  xtt03    xtt04
[oracle@erp out]$ ls
out01 out02 out03 out04
```

下表显示了xtt.properties文件中tablespaces=参数的示例。

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_MEDIA, APPS_TS_NOLOGGING, APPS_TS_QUEUES, APPS_TS_SEED…

为防止在并行运行时xttdriver.pl删除任何现有的数据文件备份副本，请在中注释掉以下一行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;
   }
```

用于xttdriver.pl对源系统进行 RMAN 备份。

如果源系统是 Oracle RAC，则可以在每个 Oracle RAC 实例上同时运行它。

xttdriver.pl 的输出存储在 TMPDIR 环境变量中。使用--backup选项运行每个xttdriver.pl命令，然后使用该--debug选项打开调试模式。

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

在本示例中，<nn>表示表空间组。如果有四个表空间组，则<nn>变为01、0203、和。04

步骤 2：将完整备份副本传输到目标系统

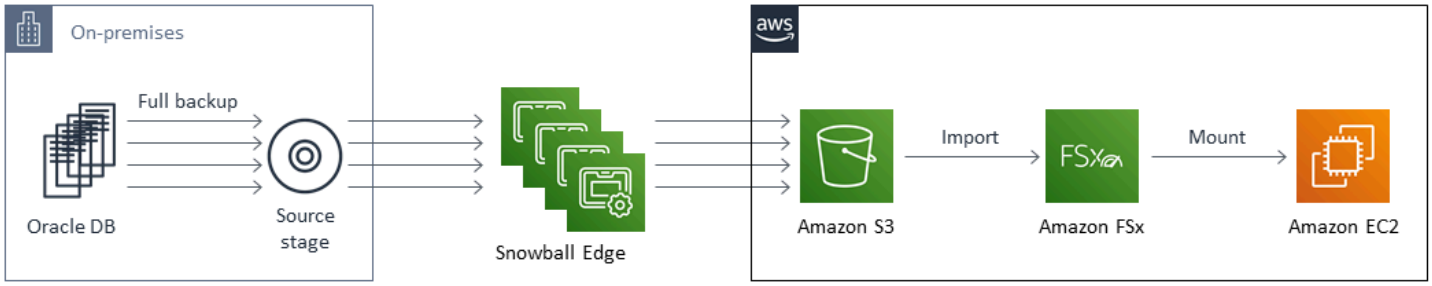
使用以下两个选项之一将备份副本传输到目标系统。

选项 1 — 使用 AWS Snowball Edge

路径：源数据库-> 源阶段-> 亚马逊 S3 AWS Snowball Edge -> FSx for Lustre

> **Note**
>
> AWS Snowball Edge 不再向新客户提供。新客户应探索AWS DataSync在线传输、用于安全物理传输AWS 的数据传输终端或 AWS Partner 解决方案。对于边缘计算，请探索AWS Outposts。

下图显示了使用 Snowball Edge 传输完整备份的情况。



Snowball Edge 是一款坚固耐用的物理存储和计算设备，可用于将大规模数据移动到其中。 AWS Snowball Edge 有助于克服大规模数据传输中可能遇到的挑战，包括传输时间长、可用带宽不足和安全问题。

整个数据文件备份副本由 Snowball Edge 传输到 Amazon S3。如果源系统大小超过 100 TB，则必须使用多台 Snowball Edge 设备来缩短传输时间。

适用于 Lustre 的 Amazon FSx 与 Amazon S3 深度集成。您可以在几分钟内为 Lustre 创建链接到 S3 存储桶的 FSx for Lustre 文件系统。当链接到 Amazon S3 数据存储库时，FSx for Lustre 文件系统会以文件形式透明地呈现 Amazon S3 对象。在真实环境中，FSx for Lustre 的性能取决于其存储容量、每个文件的大小以及文件在文件系统中的分布情况。当将 FSx for Lustre 用作目标阶段存储时，您无需将表空间备份副本从亚马逊 S3 恢复到亚马逊弹性块存储 (Amazon EBS) Elastic Block Store 或亚马逊弹性文件系统。

1. 将 S3 存储桶导入 FSx for Lustre。

   使用 Snowball Edge 将源舞台区域（例如`src_backups`）传输并存储到 S3 存储桶（例如）中`s3-src-backups`。

   创建 FSx for Lustre 文件系统（`dest_backups`例如）作为数据文件备份副本的目标舞台区域。

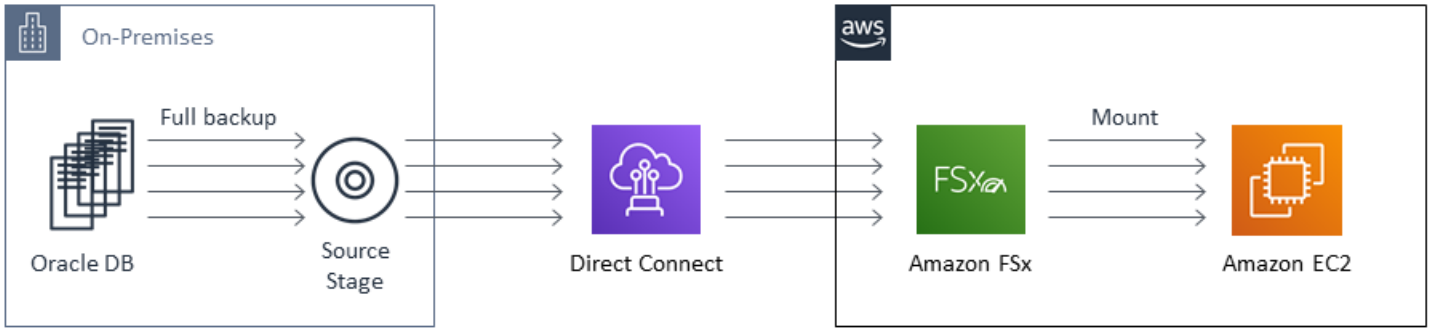
   在目标系统 (Amazon EC2) 上安装开源 Lustre 客户端。有关更多信息，请参阅适用于 Lustre 的 Amazon FSx 用户指南。

   安装 Lustre 客户端后，将 FSx for Lustre 文件系统挂载到目标系统 (Amazon EC2)。
2. `res.txt`从源`$TMPDIR`上传输到`$TMPDIR`目的地。

选项 2-使用 AWS Direct Connect

路径：源数据库-> 源阶段- AWS Direct Connect > fsX for Lustre

下图显示了使用传输完整备份的情况 AWS Direct Connect。



Direct Connect 使您能够在数据中心、办公室或托管环境之间创建专用网络连接，以及 AWS。这些私有网络连接可降低网络成本、提高吞吐量并提供一致的体验。

您可以直接将数据文件备份副本从源系统传输到装载 Amazon FSx for Lustre 文件系统的目标系统 (Amazon EC2)。如果您能容纳的高带宽，则此选项比 Snowball Edge 选项快。 Direct Connect

在 Snowball Edge 传输数据文件备份副本后 Direct Connect，你可以在目标舞台区域的 FSx for Lustre 文件系统中看到它们。

步骤 3：恢复完整备份副本并转换数据文件

恢复完整备份副本并将数据文件转换为目标系统字节序格式。要缩短还原和转换持续时间，请运行带有每个表空间组`--restore`选项的`xttdriver.pl`命令。

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

步骤完成后，数据文件将放置在目标系统上文件中定义的 dest_datafile_location 中`xtt.properties`。

第 2 阶段 — Roll-forward 阶段（源数据库保持在线）

您可以根据需要多次重复前滚阶段，以将目标数据文件捕获到源数据库。

向前滚动阶段包括以下步骤。

1. 从源数据库创建增量备份。
2. 将备份传输到目标系统。

3. 将备份转换为目标系统字节序格式。
4. 将备份应用于转换后的目标数据文件副本以将其向前滚动。

每次连续的增量备份都应花费更少的时间，并且会使目标数据文件副本与源数据库保持一致。

步骤 1：并行进行增量备份

并行备份源系统上正在传输的表空间组的增量备份。

此步骤将为文件中列出的所有 tablespaces= 创建增量备份。`xtt.properties`

如果可以在源系统上激活 “块更改跟踪” 功能，则可以大大缩短增量备份的时间。

以下命令会自动识别完全备份后的下一个 SCN。

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

步骤 2：将增量备份和 res.txt 文件传输到目标系统

如果您有足够的带宽，则可以通过使用来缩短传输持续时间 Direct Connect。

在`src_scratch_location`和之间传输增量备份`dest_scratch_location`。在源系统和`$TMPDIR`目标系统之间`$TMPDIR`传输`res.txt`文件。

如果您进行多个增量备份，则必须在最后一次增量备份之后复制该`res.txt`文件，然后才能将其应用于目标系统。

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

步骤 3：转换并应用增量备份

将增量备份转换为目标系统字节序格式，并将增量备份应用于目标系统上的目标数据文件副本。

在本指南中，假设表空间组是四个。使用每个表空间组的`--restore`选项运行每条`xttdriver.pl`命令。

在此步骤中，Oracle XTTS 实用程序会启动要重新启动的目标数据库。要并行运行该命令，必须在 Perl 脚本中使用以下自定义设置。`xttdriver.pl`

1. 注释掉以下几行：

   - 4867 号线： `my $outputstart = `sqlplus -L -s \"/ as sysdba\" \@xttstartupnomount.sql`;`
   - 4868 行： `checkError ("Error in executing xttstartupnomount.sql", $outputstart);`
   - 第 4992 行： `my $outputstart = `sqlplus -L -s \"/ as sysdba\" \@xttdbopen.sql`;\`

2. 将目标数据库设置为NOMOUNT状态。

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

3. 并行执行增量备份的前滚操作。

```
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. 向前滚动所有增量备份后，将目标数据库设置为OPEN状态。

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

此时，您可以重复第 2 阶段，直到源数据库和目标数据库上的 SCN 足够接近。您必须根据给定的目标停机时间决定是进入第 3 阶段还是重复第 2 阶段。

为了缩短第 3 阶段（传输阶段）的停机时间，可以在将源数据库设置为USER之前导出和导入非分段对象的元数据，例如FUNCTION、、和。PACKAGE PROCEDURE READ ONLY

如果源数据库中的数据库对象数量非常大（数十万），则导出和导入元数据将需要几个小时。在这种情况下，可以考虑导出元数据。

导出基于非分段的对象是在源系统中 read/write 运行的。然后，必须先将对象导入目标系统，然后才能将源系统设置为READ ONLY。此时，必须保持PACKAGEPROCEDURE、和等数据库源对象FUNCTION不变。

这是转储参数文件的一个示例，该文件用于导出非基于分段的对象的元数据，包括、USER、PACKAGE_SPECPACKAGE_BODY、PROCEDURE和。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
```

使用以下命令导出元数据。

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

这是用于导入非分段对象元数据的转储参数文件的示例。

```
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,
…..
```

目前，目标系统TEMP上除了SYSTEM、SYSAUXUNDO、和之外没有其他表空间。因此，必须重新映射所有要导入到SYSTEM表空间的对象。

使用以下命令导入元数据。

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

现在，您可以在目标数据库上看到创建的对象。

```
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
…..
```

除了SYSTEM、、SYSAUX和之外没有其他表空间。UNDO TEMP创建USER对象时使用默认表空间USERS作为默认表空间。在第 3 阶段，将创建可传输表空间，然后使用原始默认表空间更改 USER 对象。

阶段 3-传输阶段（源数据库为只读）

在此阶段，源系统变为只读。通过应用最终的增量备份，可以使目标系统上的数据文件与源系统保持一致。然后，从源系统导出对象元数据并将其导入目标系统。

步骤 1：将源数据库中的表空间设为只读

使用 SYSDBA，在源系统READ ONLY上传输所有表空间。

为了减少停机时间，您可以同时运行以下两个步骤。

步骤 2：创建最终的增量备份

在源系统上，创建正在传输的表空间的最终增量备份。

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

此步骤会返回错误，例如 “ORA-20001：TABLESPACE (S) 为只读”；该错误是预料之中的，您可以放心地将其忽略。

步骤 3：导出元数据

从源数据库导出可传输表空间的元数据。

这是用于导出可传输表空间元数据的参数文件示例。

```
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
```

此外，如果源系统有许多表和索引，则可以在导出过程中排除它们的统计信息，从而节省导出时间。导入可传输表空间后，将统计数据导入目标系统。

在运行之前expdp，请创建一个数据库目录，用于存储源系统上的转储文件。

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

以下两个步骤是使用 RMAN 增量备份实现跨平台可传输表空间的最后步骤。这些步骤必须按顺序执行。

步骤 4：传输文件并应用最终的增量备份

将最终的增量备份和导出转储文件传输到目标系统，进行转换并应用最终的增量备份。

用于 Direct Connect 将最终增量备份副本和 res.txt 文件传输到目标。您可以使用 VPN 连接，但如果有足够的带宽， Direct Connect 则使用可以显著减少停机时间。

要恢复最终的增量备份，请在目标系统上使用--restore选项对每个表空间组运行以下命令。

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

步骤 5：导入对象元数据

使用 Oracle 数据泵将对象元数据导入目标系统。运行以下命令以获取目标系统上transport_datafiles=参数的数据文件列表。

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

每当你运行前面的命令时，你都会得到带有transport_datafiles=参数的xttplugin.txt文件。将所有xttplugin.txt文件合并为transport_datafiles=一行，然后将数据文件列表放入导入元数据的参数文件参数中。transport_datafiles

以下代码片段显示了用于在目标系统上导入可传输表空间的参数文件。

```
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'…..
```

在运行之前impdp，创建一个指向导出转储文件位置的数据库目录。

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

第 4 阶段 — 验证传输的数据

步骤 1：检查表空间是否损坏

在此步骤中，传输的表空间在目标数据库中是只读的。您可以通过运行 RMAN 命令来验证表空间是否有效，如下所示。

```
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
```

此外，您还可以检查对象的数量，例如表、索引、同义词、视图和包对象。

步骤 2：将所有传输的表空间放在目标数据库中 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
```

结论

在本指南中，您学习了如何通过使用 Oracle 跨平台可传输表空间和 RMAN 增量备份、 AWS Snowball Edge和 Amazon for Lustre 来最大限度地减少停机时间。 AWS Direct Connect FSx

将 Oracle 数据库从本地迁移到本地时 AWS，请考虑以下变量：

- 所需的网络带宽 Direct Connect
- for Lustre FSx 的容量
- Oracle 数据库的元数据大小
- 增量备份大小

使用本指南中推荐的方法，从 Unix 系统上运行的 100 TB Oracle 数据库迁移到时 AWS，可以将停机时间缩短到大约 10 小时。

文档历史记录

下表描述了本指南的重大更改。如果您希望收到有关未来更新的通知，可以订阅 RSS 源。

变更	说明	日期
–	更改了简介和概述中对异构的提法。	2021 年 7 月 14 日
–	初次发布	2021 年 3 月 2 日

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。