Very Large Database (VLDB) Backup & Recovery Best Practices

An Oracle White Paper
July 2008

ORACLE

Very Large Database Backup & Recovery Best Practices

Introduction 4
Key Database Protection Requirements 4
Determine Recovery Requirements 4
Analyze and Identify the Cost of Unavailable Data 4
Assess Tolerance for Data Loss 5
Assess Total Recovery Time 5
Determine Recovery Window 5
Architect Backup Environment 5
Determine Backup Destination: Disk, Tape, Virtual Tape Library 6
Evaluate Backup Solutions 6
Oracle Recovery Manager 7
Oracle Secure Backup 7
Oracle Flashback Technologies 7
Oracle Data Guard 7
Third Party Snapshot, Split Mirror, Continuous Data Protection Solutions 8
Plan Data Layout 9
Read-only Tablespaces 9
Table Compression 9
Develop Backup Strategy 10
NOLOGGING Considerations 12
Other Backup Strategy Considerations 12
Multi-cycle Full Backup 12
Tablespace-level Backup 13
Multi-section Backup 13
Backup Compression 14
Guaranteed Restore Points 14
Develop Recovery Strategies 15
Media Failure 15
Block Corruption 15
User Error 16
Disaster Recovery 16
VLDB Standby Database Creation 16
Starbucks VLDB Backup & Recovery Case Study 16
Background 17
Backup Solution 20
Conclusion 23

References 24
Appendix A - Starbucks RMAN Scripts 25
`Daily_incr_bkup_disk` – Daily Backup to FRA 25
`Daily_full_bkup_tape` – Daily Backup of Incremental and Archived Log Backups to Tape 26
`Weekly_full_bkup_tape` – Weekly backup of FRA to Tape . 26
`Monthly_full_bkup_disk` – Monthly Full Backup to Disk ... 26

INTRODUCTION

Over the years, the definition of a Very Large Database (VLDB) has radically changed – whereas 100 GB might have been considered very large 20 years ago, in today's environment, a new database deployment in a large company might *start* at 1 TB. In fact, it is not uncommon to find databases in the tens to hundreds of TB's and even PB's, serving traditional data warehouse and increasingly, OLTP activities [1]. The dramatic rise in database sizes, particularly over the last few years. imposes unique challenges for database administrators, who are expected to protect large volumes of data in a timeframe that is ever shrinking due to stringent service level requirements. This paper outlines best practices for meeting the challenges associated with backing up and recovering VLDBs to protect against media corruptions, data failures, as well as human and application errors. Finally, Starbucks shares its hands-on experiences in the design and implementation of a VLDB backup and recovery strategy.

KEY DATABASE PROTECTION REQUIREMENTS

The key VLDB backup and recovery questions which this paper addresses are:

- How to backup/recover in a satisfactory timeframe?
- How to recover at an application or business object level?
- How to protect data in 24x7 environment?
- How to manage software, infrastructure, operational costs for data protection/disaster recovery?

With these questions in mind, an outline for thinking through backup and recovery design and deployment is now presented.

DETERMINE RECOVERY REQUIREMENTS

A backup environment and strategy serves no good unless it can fulfill the stated recovery requirements. So, the first task is to assess these requirements.

Analyze and Identify the Cost of Unavailable Data

What is the business impact of lost revenue and productivity due to unavailable databases or portions of the database? Quantifying these costs upfront paves the

way for justification of appropriate hardware, storage, and software expenditures to meet business-mandated service levels.

Assess Tolerance for Data Loss

Recovery Point Objective (RPO) is the degree to which data loss can be tolerated, e.g. 24 hours, 1 hour, zero data loss. Lower tolerance for data loss will require more frequent backups and/or additional hardware and storage infrastructure. For example, if zero data loss is required, relying on more frequent tape backups will not be sufficient; instead, solutions such as Oracle Data Guard should be considered.

Are some database objects (e.g. tables, tablespaces) less critical than others? If so, these objects could potentially be backed up less frequently than others, thus reducing overall backup time and storage.

Is point-in-time recovery required? If so, the database needs to be in archived log mode, and additional space on production storage and backup media must be provisioned for these logs and their backups.

The responses here play a key role in dictating requirements for the backup environment and strategy.

Assess Total Recovery Time

Recovery Time Objective (RTO) is the acceptable recovery time for the whole database and for subsets of data (if applicable). Note that total recovery time not only includes restoring backups followed by media recovery, but also includes time to identify the problem and plan appropriate recovery, in addition to any hardware, storage, and Oracle software restoration time.

To achieve very low recovery time objectives for VLDBs, relying on traditional tape backups will not suffice. Therefore, solutions such as disk backup and Oracle Data Guard should be considered, along with their required hardware and storage infrastructure. Again, the cost to the business, while data is unavailable, will drive justification of additional infrastructure.

Determine Recovery Window

A recovery window ensures that a sufficient number of backups are maintained to provide point-in-time recovery within the specified timeframe. The recovery window is typically set with a backup retention policy. The longer that backups must be retained with a regular backup schedule, the more disk and/or tape will typically be needed.

ARCHITECT BACKUP ENVIRONMENT

At this point, the documented recovery requirements now drive the decision on a backup infrastructure (hardware, storage) and backup strategy (tools, procedure). If the optimal infrastructure and strategy are not accomodated by the available budget,

the solutions can be iteratively scaled back and reviewed, assuming that any risks to recovery requirements are communicated to the business owners.

Determine Backup Destination: Disk, Tape, Virtual Tape Library

Backups are commonly made to disk, tape, and/or virtual tape library (VTL). VTL allows backups to be written to a storage array via a standard tape interface; thus, one can achieve the speed and reliability of disk backup, without having to change the existing tape backup interface or procedure. Backups on VTL are then migrated to tape for long-term archival by the media manager. A high-level comparison of these options follows:

	Disk	Tape Library	Virtual Tape Library (VTL)
Compatibility, Provisioning	- OS, drivers, storage must be compatible - Requires ongoing capacity monitoring	- Well-known interface across heterogeneous systems - Tapes easily added, as needed	- Emulates standard tape devices - Reduced administrative complexity versus disk
Performance	- Fast, random I/O access	- Slower, sequential access	- Fast, random I/O access
Disaster Recovery	- Optional block-level mirroring	- Offsite tapes for long-term archival and/or DR	- Optional file-based replication with deduplication
Cost	- Price/capacity starts at few dollars/GB (ATA)	- Best price/capacity, e.g. LTO-3 ($60 list) holds 800 GB compressed	- VTL license + disk cost (appliance-based)

Figure 1. Comparison of Disk, Tape Library, and Virtual Tape Library

As can be seen, disk and VTL offer the best performance due to inherent random I/O access, but at a higher cost. Whether the higher cost can be justified is again dictated by the business-mandated service levels and available budget. For example, rather than deploying VTL backups for all VLDBs, which might be cost prohibitive, prioritize each database's criticality and leverage VTL for the most critical ones and tape for all others. Another method to reduce overall disk cost, is to use both disk and tape backup for the most critical tablespaces and use only tape backup for all others.

Evaluate Backup Solutions

In tandem with selecting the appropriate backup infrastructure, various backup solutions should be evaluated, both in terms of functionality and cost. The following details a few of the most popular solutions.

Oracle Recovery Manager

Recovery Manager (RMAN) is the Oracle-native solution for creating and managing physical database backups and automating database restore and recovery [2]. RMAN offers multiplexed, parallel, block-validated full and incremental backups. It supports data file, tablespace, and database backup and recovery, in addition to block media recovery. With Oracle Database 10*g*, RMAN can leverage an Oracle-aware disk backup strategy with the Flash Recovery Area, a disk location where all recovery-related files (e.g. backups, archived logs) are stored and managed by Oracle (e.g. automatic deletion of obsolete files and files backed up to tape, when additional space is required). Backups can be taken seamlessly in a RAC environment and/or Data Guard, where backup operations can be offloaded to specific nodes or physical standby databases. RMAN is integrated with Oracle Secure Backup for tape backup and supports all leading 3rd party tape backup products.

Oracle Secure Backup

Oracle Secure Backup (OSB) provides low-cost, centralized tape backup management for database & file systems [3]. Through native integration with RMAN, OSB offers the fastest database backup to tape on the market via elimination of unused blocks and committed undo during backup. OSB is affordably priced at just $3,500/tape drive with no additional component or option costs.

Oracle Flashback Technologies

Flashback Technologies is a broad suite of fast user error analysis and correction tools at the row, transaction, table, and database level [4]. Flashback operates by undoing changes to the specific data, relative to the desired point-in-time versus performing traditional restore and recovery from disk or tape backup, which can take several hours or even days for a VLDB. Flashback Database provides built-in, optimized Continuous Data Protection (CDP) for the database, similar to a fast 'rewind' capability. Other features such as Flashback Table and Flashback Transaction provide granular point-in-time recovery at the table level and automated transaction backout.

Oracle Data Guard

Oracle Data Guard is software available with the database, that creates, maintains, and monitors one or more standby databases to protect enterprise data from failures, disasters, errors, and corruptions [5]. Data Guard maintains these standby databases as synchronized copies of the production database. These standby databases can be located at remote disaster recovery sites thousands of miles away from the production data center, or they may be located in the same city, same campus, or even in the same building. If the production database becomes unavailable because of a planned or an unplanned outage, Data Guard can switch

any standby database to the production role, thus minimizing the downtime associated with the outage, and preventing any data loss.

Third Party Snapshot, Split Mirror, Continuous Data Protection Solutions

Third party storage solutions are generally not aware of Oracle block formats and structures, so cannot provide the granularity of recovery as the aforementioned Oracle technologies (e.g. recovery of individual tables). However, being optimized for storage-native operations, these solutions can provide fast 'backup' methods, albeit with additional licensing and storage costs.

The most popular third party techniques are snapshot, split mirror, and continuous data protection (CDP). Snapshot technology provides fast point-in-time copies of the database using incremental storage block tracking; however, the availability of the copy typically depends on the availability of the production array. For disaster recovery purposes, the copy should be fully re-created on a separate array. Similarly, disk mirroring technology allows a block-level mirrored volume to be quickly 'split' onto a separate array to create a point-in-time copy of the database. RMAN backups can be configured on snapshot and split mirror copies, but requires additional coordination with storage operations [6]. Supported backup, restore, and recovery of third party snapshots is documented in Metalink Note 604683.1 [7]. Note that Oracle fully supports offloading backups to a physical standby database and these backups can be used to restore any database in the Data Guard configuration [8] [9].

CDP solutions extend point-in-time snapshot capabilities to a continuous snapshot, for fast recovery to any point-in-time within a specified time period. Similar to snapshots, CDP leverages copy-on-write or allocate-on-write technology to track changed blocks.

Flashback Database offers distinct advantages over snapshot and CDP solutions for the database. When a block changes occurs, snapshots typically require an additional block read as a precursor to copying the storage block to the snapshot volume. Flashback Database does not require this additional block read before it copies a before-change block to the Flashback log. CDP solutions incur additional I/O overhead to maintain a "change journal" of all database files (e.g. data file, redo, control file), whereas Flashback Database only tracks data file block changes. In addition, CDP solutions must track each and every storage block change, whereas Flashback Database only logs an Oracle block change every 30 minutes, regardless of the number of changes to the block – this optimization is only possible with Oracle, where log-based recovery is leveraged during the Flashback operation itself to reach the desired point-in-time within the 30 min interval. Again, these space and I/O optimizations are only possible since Flashback Database is fully Oracle-aware.

Plan Data Layout

Backup time and space usage can be optimized through efficient data layout, before the first backup is ever taken. Two common techniques are read-only tablespaces and table compression.

Read-only Tablespaces

The advantage of a read-only tablespace is that it only needs to be backed up once within a defined tape recycling period, rather than with every backup job. If a data warehouse contains five years of historical data and the first four years of data can be made read-only, then a regular backup of the database will only backup 20% of the data, until the read-only backup tapes are recycled. This can dramatically reduce the amount of time required to back up the data warehouse.

Many data warehouses store their data in tables that have been range-partitioned by time. For example, a retailer may accept returns up to 30 days beyond the date of purchase, so sales records could change frequently during this period. However, once data ages past a certain date after 30 days, it becomes relatively static.

By leveraging partitioning, users can make the static portions of their data read-only. Consider the following processes to do so:

- Implement a regularly scheduled process to move partitions from a read-write tablespace to a read-only tablespace when the data matures to the point where it is entirely static.
- Or, create a series of tablespaces, each containing a small number of partitions and regularly modify one tablespace from read-write to read-only as the data in that tablespace ages.

Table Compression

Table compression is another method to reduce overall database size and consequently, backup size. Table compression was first introduced in Oracle9*i* to allow data to be compressed during bulk load operations such as direct path load, CREATE TABLE AS SELECT operations, etc. This form of compression is ideally suited for data warehousing environments where most data is loaded in the database via batch processes. Additionally, there is no measurable degradation when reading compressed data, as Oracle can read the compressed block directly without requiring uncompression. With Oracle Database 11g, OLTP Table Compression was introduced to further minimize the system impact due to updates on compressed tables, through batch compression of the block following a series of writes, versus having to re-compress the block after each write. More information can be found in the Oracle Advanced Compression white paper [10].

Develop Backup Strategy

Assuming RMAN is utilized for database backups, the following backup strategies can be evaluated, based on the required recovery time objective and available disk and/or tape resources.

1. Level 0 (full) + Fast Incremental Backups

Appropriate for VLDBs that are:

- Able to tolerate hours/days for restore & recovery
- Limited to tape backup-only environments
- Changing on a low-medium frequency between backups

In this strategy, weekly level 0 and nightly cumulative incremental backup sets are taken to tape, with optional RMAN or tape device compression. Backup sets are space-optimized, leveraging unused block compression and committed undo elimination, in addition to BZIP and ZLIB-based compression, as discussed below.

Only the changed blocks are read and written with the fast incremental backup, which are well-suited for VLDBs with low-medium frequency of changes. Archived logs are backed up and retained on-disk, as needed. As shown below, recovery is possible to any point-in-time between the level 0 backup and current time, using archived log and incremental backups.

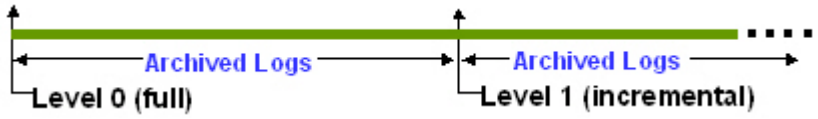

Figure 2. Level 0 + Level 1 Fast Incremental Backup Strategy

2. Image Copy (full), Fast Incremental Backups, Incrementally Updated Backup

Appropriate for VLDBs that have:

- Maximum tolerance of no more than several hours for restore & recovery
- Provisioned disk backups for full database image copy or image copy of most critical tablespaces

In this strategy, an initial level 0 image copy is made to disk, followed by nightly incremental backups. The image copy is rolled forward with the most current incremental, to produce a new on-disk full backup, on a regular basis (e.g. daily/weekly/monthly). Archived logs are backed up and retained on-disk, as needed. As shown below, fast recovery from disk is possible to any point-in-time between the initial image copy backup or rolled forward copy (whichever was created most recently) and current time, using archived logs and incremental backups. Image copies can be restored from disk or quickly used by the production

database via the RMAN `SWITCH` command (i.e. no restore time) and applying needed redo for recovery. Image copies and incrementals on disk are archived to tape, as needed, if there is a requirement to recover to points-in-time older than the on-disk copy. Note that image copies require the same amount of disk space as the production data files (minus temp files), in contrast to backup sets as described in the previous strategy. Therefore, plan storage requirements accordingly, including reviewing documentation on how to size the Flash Recovery Area [11].

Figure 3. Level 0 + Level 1 Fast Incremental + Incrementally Updated Backup Strategy

3. Data Guard + Level 0/Fast Incremental Backups

Appropriate for VLDBs that have:

- Maximum tolerance of no more than several minutes of recovery time, in event of any failure.
- Budget to allocate symmetric hardware and storage for physical standby database.
- Shared tape infrastructure between primary and standby database sites.

Full and incremental backups can be offloaded to a physical standby database and used for restore at primary or standby database, as shown below. Alternatively, backups can be taken at each database for optimal local protection. Refer to OTN and Oracle documentation for procedural details [8] [9].

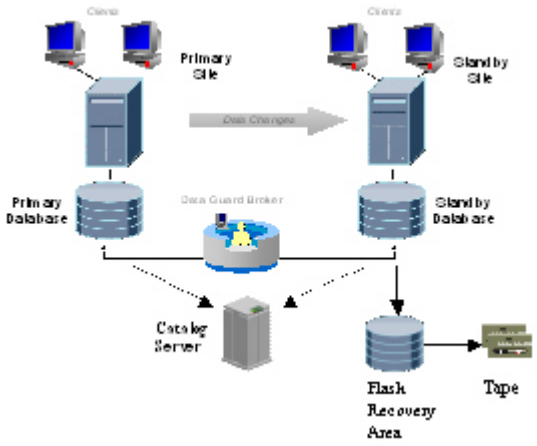

Figure 4. RMAN Backup in Data Guard Environment

4. Maintain Extract, Transform, Load (ETL) Files

Appropriate for VLDBs that:

- Do not require point-in-time recovery
- Do not have space for archived logs or incremental backups
- Run widespread NOLOGGING for performance reasons
- Are confined to controlled batch loads

Data warehouses are commonly used to consolidate data from various sources. The ETL (or Extract, Transform, Load) process starts with flat files from the original sources loaded into the warehouse as base tables, and then transformed into analysis and report application-ready tables. An ETL job would contain the flat files plus load and transformation procedures.

In this strategy, a full backup is periodically taken, and *n* ETL jobs are maintained, where *n* is number of jobs between full backups. As shown below as 'X's, recovery is only possible to the full backup and the ETL job times. The recovery procedure is to restore last full backup and run needed ETL jobs. Note that this procedure will not capture changes outside the ETL process and is less automated for recovery than previously discussed RMAN strategies.

Figure 5. ETL Load Maintenance Backup Strategy

NOLOGGING Considerations

VLDBs, and in particular data warehouses, may use NOLOGGING operations to increase performance and save space. However, recovery to a point-in-time within a NOLOGGING time period will result in corrupt blocks. Therefore, the best practice is to take an incremental backup when NOLOGGING finishes to preserve recoverability. As shown below, recovery is possible to any point in time, except for NOLOGGING periods. Given that the backup must be taken when all NOLOGGING operations complete, this strategy is better suited for controlled batch load environments.

Figure 6. Level 0 + Level 1 Fast Incremental Backups in NOLOGGING Environment

Other Backup Strategy Considerations

Multi-cycle Full Backup

Some large databases may not be able to practically take a full backup within a single backup window, e.g. midnight-6 am. To help remedy this situation, RMAN

allows a full backup to be divided across multiple windows and tracks which files are backed up and which still need a backup.

For example, the RMAN command:

```
BACKUP DATABASE NOT BACKED UP SINCE 'SYSDATE-3' DURATION
06:00 PARTIAL MINIMIZE TIME;
```

will attempt to complete as much of the full backup as possible, as fast as possible, within the next 6 hour window (DURATION 06:00). When the backup has run for 6 hours, it will terminate and exit successfully (allowed by PARTIAL). When the command is run again for the next day's backup window, RMAN will start with the last file not backed up from the previous backup cycle (NOT BACKED UP SINCE 'SYSDATE-3') and continue the backup for 6 hours. If not all files are backed up in this day's window, they will be the first to be backed up on the following day. With this method, the entire database can be backed up over the course of a few days (in this case, 3 days). Note that since backups are spread across multiple days, there is a possibility that backup files may be spread across more tapes versus backing up within a single (albeit longer) window to a confined set of tapes. Consequently, full database restore and recovery time could be impacted as more tapes need to be retrieved and mounted to restore all needed data files, in addition to applying more archived logs during recovery.

Tablespace-level Backup

In some Oracle environments, certain tablespaces are less critical than others, due to the nature of the applications. Backup time can be reduced by backing up those tablespaces less frequently than others. In addition, index tablespaces may be backed up less frequently than data tablespaces. This strategy can also improve restore time, e.g. by grouping critical tablespaces in a separate backup when going to tape, one or more of the tablespaces can be restored faster from this special backup rather than having to spin through a full backup on tape to locate them. Note that a tablespace that has not been backed up in a long time will require application of more archived logs during recovery, than a more recent backup.

As discussed previously, tablespace backups allow backup of critical tablespaces to disk, while less critical ones can be backed up directly to tape, thus devoting higher performant disk for critical recoveries. Refer to the Oracle Backup and Recovery User's Guide for examples of tablespace backups.

Multi-section Backup

With the rapid growth of databases, one challenge is to manage larger datafiles and tablespaces. In fact, with the introduction of `BIGFILE` tablespaces in Oracle Database 10g, datafiles can now be sized in excess of 1 TB+. In-line with this trend, in Oracle Database 11g, RMAN offers multi-section backup to support intrafile parallel backup. This feature allows one data file to be divided into user-specified 'sections', so that each section can be backed up in parallel on separate

channels. This method is best for databases composed of a few data files that are significantly larger than the rest, versus a database composed of large number of small data files. This is also optimal for databases where there are fewer number of large data files than the number of available tape drives. An initial recommendation for section size = (average data file size / # of channels). Channels can be incrementally added to scale up backup performance, depending on hardware and tape drive limits.

Backup Compression

Keeping backups as small and efficient as possible is especially important for large databases. Oracle provides a number of space optimizations, in contrast to third party backup solutions. RMAN automatically eliminates unused blocks from full backup sets and with Oracle Database 11*g*, committed undo is also eliminated. This can make a dramatic difference for VLDBs, e.g. a 1 TB database with only 100 GB used space will result in a ~100 GB backup, not 1 TB. As another example, if a 1 TB table is dropped and purged, successive full backups will not incur reads or writes of the 1 TB worth of unused blocks, whereas third party storage solutions would need to continue to backup those blocks.

Oracle Database 11*g* RMAN offers Fast Backup Compression, providing compression performance 40% faster than backup compression in Oracle Database 10*g*. Note that a trade-off is that Fast Backup Compression has a slightly lower compression ratio versus 10*g*-style compression. In Oracle Database 11*g*, customers can choose either compression strategy, that best suits their needs.

Guaranteed Restore Points

As discussed previously, Flashback Database is a fast, continuous point-in-time recovery method to repair widespread logical errors. Flashback Database relies on additional logging, called flashback logs, which are created in the Flash Recovery Area and retained for a user-defined period of time. These logs track the original block images when they are updated.

When a Flashback Database operation is executed, just the block images corresponding to the changed data are restored and recovered, versus traditional data file restore where all blocks from the backup need to be restored before recovery can start. Flashback logs are generally created in proportion to redo logs.

For very large and active databases, it may be infeasible to keep all needed flashback logs for continuous point-in-time recovery. However, there may be a need to create a specific point-in-time snapshot (for example, right before a nightly batch job) in the event of logical errors during the batch run. For this scenario, guaranteed restore points (GRPs) can be created without enabling flashback logging.

When a GRP is created this way and a block changes, only the first before-image for that block is ever written to the flashback logs, to maintain the state of the block at that GRP. This is in contrast to enabling flashback logging, where a

before-image of the same block could be logged several times as it changes over time, with one before-image logged per 30 minute interval (as discussed previously in comparing Flashback logging against Third Party Snapshot and CDP Solutions). Thus, if a workload modifies a large number of the same blocks repeatedly, creating a GRP without enabling flashback logging will save flashback log space.

For example, a GRP can be created, then followed by a NOLOGGING batch job. In event of batch job issues, Flashback Database to the GRP will quickly undo the batch job changes. When the GRP is no longer needed, it can be dropped and its corresponding flashback logs will be deleted.

Estimating flashback log space for the GRP in this scenario depends on how much of the database will change over the number of days the GRP is maintained. For example, to create and retain a GRP for 1 day where 100 GB of distinct blocks are changed during that time, plan for 100 GB worth of flashback logs. This might be in contrast to enabling flashback logging for the same database, which generates 200 GB of redo in 1 day, with a GRP created and retained for 1 day – in this case, initially plan for 200 GB worth of flashback logs, as flashback logs are typically generated in proportion to redo logs.

Note that this configuration requires that flashback logging be enabled at least once in the past. Also, the first GRP created with flashback logging disabled requires the database to be mounted. However, the following GRPs can be created with database open or mounted.

Develop Recovery Strategies

Last but not least, regularly validate backup integrity and practice recovery scenarios on identical test systems, as suggested below.

Media Failure

- Restore database files to a new storage location.
- Validate restoring database, tablespace, and data file.
    - `RESTORE DATABASE VALIDATE;`

Block Corruption

- In test environment, verify block corruption repair with Block Media Recovery.
    - `RECOVER CORRUPTION LIST;`
- Validate archived log apply with Trial Recovery, which can detect corrupt and missing logs, after restoring data files.
    - `RECOVER DATABASE TEST;`

User Error

- In test environment, verify Flashback Query, Flashback Table, Flashback Drop, and Flashback Transaction for fine-grained recovery.
- In test environment, verify Flashback Database as a viable method to repair database-wide logical errors.
- In test environment, perform tablespace point-in-time recovery (TSPITR), including recovery through table truncates or other DDL operations.
- Research corrupt or erroneous transactions with LogMiner [12]

Disaster Recovery

- Validate restoring all files to another host and storage location.
- Test switchover/failover procedures to a standby database in a Data Guard configuration.

VLDB STANDBY DATABASE CREATION

In many environments, backups are also used to create VLDB standby databases in a Data Guard configuration. Refer to the section "Alternative Method for Standby Database Instantiation using RMAN" in the RMAN & Data Guard best practices paper for additional guidance [8].

STARBUCKS VLDB BACKUP & RECOVERY CASE STUDY

The Starbucks Data Warehouse Capacity Upgrade project was started in late 2005 with the objective of creating a Business Intelligence Infrastructure that could support Starbucks rapid growth and demand for a richer reporting environment. The existing data warehousing environment utilized the hub and spoke model consisting of a central repository and several data marts. This data warehouse could only support limited end user access as most reporting was performed via data marts. The objectives of the new data warehousing environment were to build a system that could support thousands of users and allow scalability without major hardware upgrades. The system provides reporting on sales and other key performance indicators for store managers as well as provide access to a wide variety of other information for Starbucks' internal business users. A proof of concept project was commissioned to validate whether a single or multi-instance architecture would be appropriate for Starbucks's environment.

After validating the performance characteristics of RAC and non-RAC environments as well as other requirements of the new system, RAC was chosen as the architecture to support the new data warehouse. The high availability and scalability of the technology were two of the strongest factors in making the decision to proceed with RAC. Starbucks also wanted to avoid the "forklift" upgrade that occurs too commonly when a single server is outgrown.

The below discussion outlines how Starbuck's utilized RMAN for backing up RAC databases in a data warehousing environment and why it was the right decision based on the given set of requirements. It also discusses a number of questions and concerns the project team encountered when attempting to implement the strategy.

Background

The following hardware infrastructure shown in Figure 7 was chosen to support the new business intelligence environment.

Hardware Architecture – 4 Node RAC Database

- Servers – 4 CPU HP ia64 Servers 1.5 GHz CPU 16 Gb memory
- Network – Private Interconnect Infiniband Ib
- Public Network – 1000 Mbps
- Storage – SAN EMC DMX3
- ASM Data Diskgroup(s) - 16 (8 Usable) Terabytes Raid 1+0 Storage 15k 146 Gb
- ASM FRA Diskgroup - 8 Terabytes Raid 5 (3+1) 10K 300 Gb
- DB Software – Oracle 10.2.0.3 EE
- Media Mgmt – Netbackup 5.1
- Tape Hardware – 4 Scalar I2K LTO2 STG NTWK Tape Drives (400 GB compressed per tape)
- Operational (Non-Disaster) Recovery
    - RTO: < 8 Hours
    - RPO: Anytime within the last 48 hours

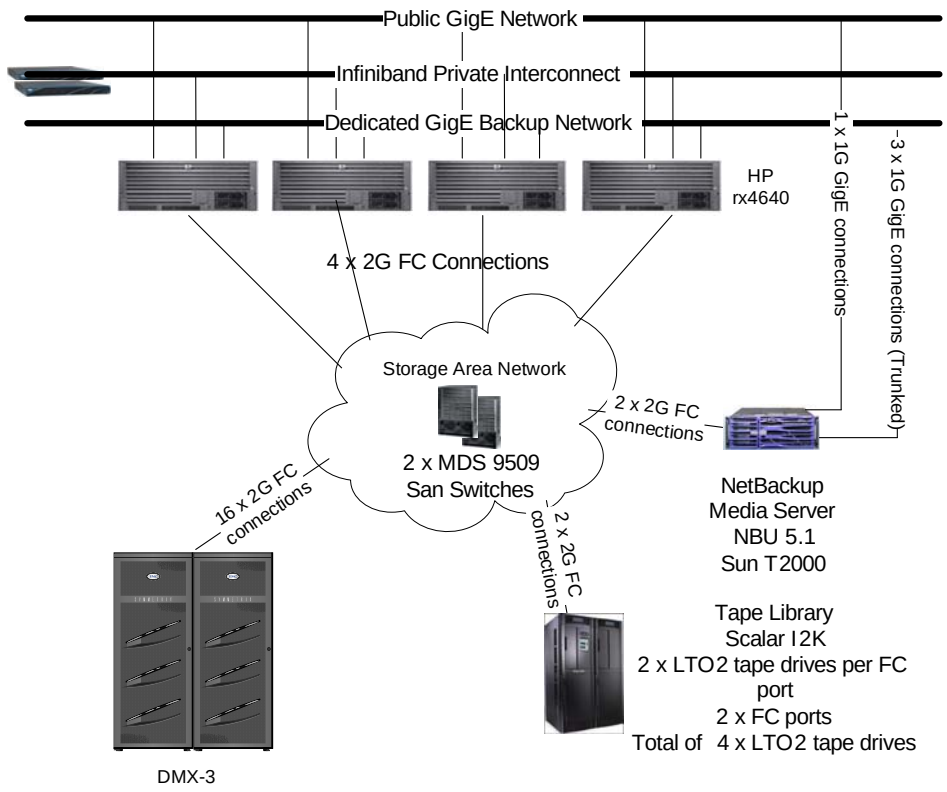

Figure 7 – Starbucks Data Warehouse Hardware Environment

Data Layout

In addition to designing the physical layout of the database for performance, there was a fair amount of deliberation as how to backup, restore, and refresh test environments that support the production system. The use of partitioning, read-only tablespaces and an archiving strategy are all keys when building a system of this size. In the case of the Starbucks Enterprise Data Warehouse (EDW), the largest fact tables consist of billions of rows, are range partitioned, and hold 39 months of data for reporting purposes. The use of a sliding window partitioning scheme was employed as a means for managing this large volume of fact data. In this case, a sliding window scheme implies partitions and tablespaces are added and dropped as time passes. Therefore, a PL/SQL package that automatically adds and drops partitions as well as adds/drops tablespaces is used to maintain the table DDL in the warehouse. Figure 8 shows an example of the Sliding Window scheme.

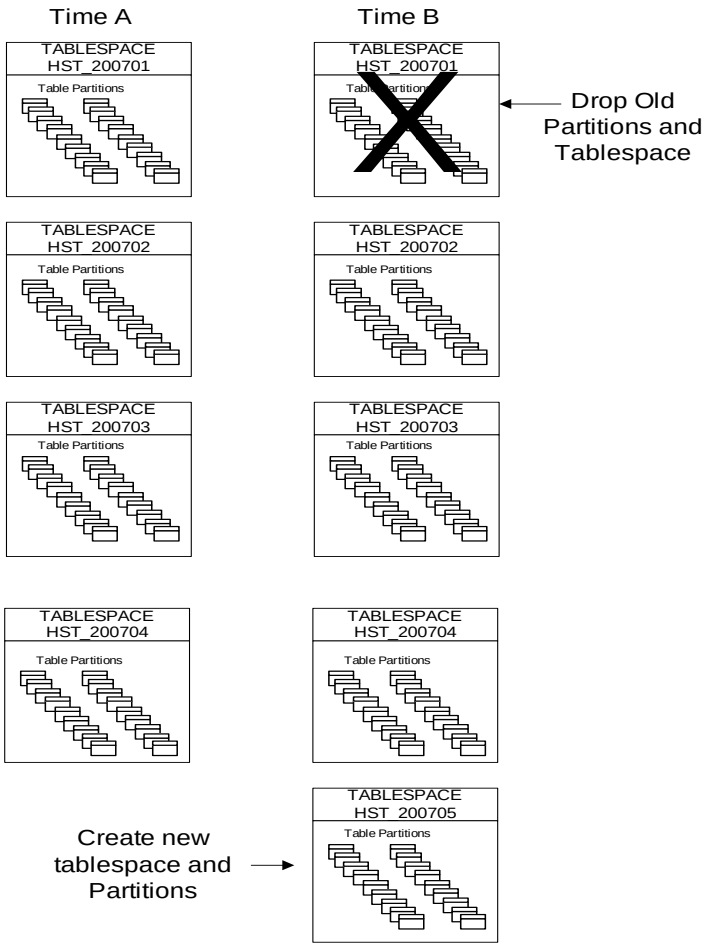

Figure 8 – Sliding Window Partitioning Scheme

As a result of the data being split at the month level over 39 separate tablespaces, the data volumes in each tablespace will average 180-200 GB in size. In the next few years, new periods (i.e. months) of data are expected to average 250-300 GB and this is believed to be a relatively manageable size from a backup and recovery standpoint. Another point of discussion in the physical design centers on the use of BIGFILE or small file tablespaces. A 16 KB blocksize had been chosen for the database, which supports datafile sizes up to 64 GB. The team opted to stay with a traditional small files approach since a 64 GB datafile still implied a manageable number of datafiles and also considering RMAN, in Oracle Database 10g, does not parallelize the backup and restore of individual BIGFILE tablepaces . Note that Oracle Database 11g addresses the parallelization of large data files with the multi-section backup feature [13]. When the database is upgraded to Oracle Database 11g, the sliding window partitioning scheme can easily support creation of new BIGFILE tablespaces, if required.

The implementation of the sliding window partitioning scheme also called for setting tablespaces containing fact data to read-only after 2 months. RMAN can automatically skip read-only tablespaces and thus optimize backup and recovery

operations. As the size of the EDW backup grows, we can take advantage of whether or not data is classified as read-write or read-only. The main goal of the physical implementation was to ensure that the EDW could scale appropriately with future data growth and that the backup and cloning-by-tablespace strategy (via RMAN DUPLICATE) could also scale.

Backup Solution

In attempting to design the optimal backup solution, the Recovery Time Objective (RTO) and Recovery Point Objective (RPO) were determined. In our case, the RTO, less than 8 hours, and RPO, anytime within the last 48 hours, could be supported by several solutions. Most notably, the high-end storage hardware chosen for the database provided the additional option of split mirror backups. Split mirror backups, by definition, are backups performed against a separate set of disks which mirror the database disks, including the Oracle home. The split mirror is brought up as a mounted instance and RMAN is used to backup the database to tape or virtual tape, since it is the only way data files residing in ASM disk groups can be backed up. The debate over database backup solutions effectively narrowed down to deciding whether to use split mirror backups or the Oracle Flash Recovery Area (FRA) for an on-disk backup solution. Beyond the typical RTO and RPO requirements, the following objectives were also considered:

1. Minimize the load on the servers and array to perform the backup tasks
2. Minimize the time it takes to run the backup
3. Keep disk space consumed by backups to a minimum
4. Ensure backup process can scale as the warehouse grows
5. Minimize recovery time

Though both solutions could satisfy the above objectives, we further weighed the strengths and weaknesses of each option and determined that the Flash Recovery Area solution provided additional benefits beyond the storage solution and tilted the decision in its favor:

Strengths

1) Lower Cost & Maintenance vs Split Mirror Backup

- Split mirror backup requires:
  - Additional split mirror software and backup server hardware licensing
  - Coordination of split mirror operations with RMAN backup scripts
  - Proper synchronization of backup server with production host

2) Built-in tablespace and block level recovery

4) Recoveries fully controlled by RMAN without requiring special storage operations

5) Physical and logical block validation

6) Incrementally updated backups, reducing need for full backups

7) Auto-reclamation of obsolete backup files per RMAN retention policy

Weaknesses

1) Load on production server while backups are run.

2) Recovery in our configuration will not be as fast as re-synching mirrors in the case of disaster recovery. Note that if identical storage to the production had been purchased for the Flash Recovery Area then performing an RMAN "switch to copy" would have been a viable option.

3) Flash Recovery Area space requirements are slightly larger than space required for split-mirror. This is mainly due to the fact that the roll-forward of the image copy requires keeping at least one incremental backup on disk.

An integrated RMAN solution was found to be simpler, more flexible and less costly to implement than Split Mirror Backups. As recommended by Oracle, we enabled Block Change Tracking to help minimize the incremental backup time. In reading various articles on OTN, RMAN documentation, and testing various solutions, an incrementally updated backup strategy was implemented.

Disk (FRA)

1. Monthly (Level 0) Image Copy backups to FRA. Monthly image copy backups are taken to check all production data file blocks for physical corruptions and ensure that full backups can be successfully created per current operations guidelines. The future plan is to reduce the number of full backups. The image copy space was initially estimated to be equal to the production database size minus temp files. Of course, the image copy will grow as the database grows, so the estimates are periodically reviewed and disk is added to the FRA, as needed.
2. Daily Level 1 Incremental Backups. Two incremental backups are always maintained in the FRA, to satisfy the roll forward strategy.
3. Daily Incremental Update (i.e. roll forward) of Image Copy for the rest of the month with 'sysdate – 2'.

Tape

1. Weekly "Backup Recovery Area" (FULL)
2. Rest of the week "Backup Backupset ALL"

After deciding on an RMAN-only backup solution, we outlined other questions and concerns raised during the design process:

Question: How could the space requirements of the Flash Recovery Area be minimized?

Secondary storage was not available for this system since other SANs were consumed by other projects and systems. This drove the decision to find an RMAN option which minimized space requirements. In particular, there was insufficient disk space to retain two database image copies on disk at the same time, which would occur right after each monthly full backup. The solution to minimizing the FRA space was to remove the existing disk backup prior to running the next full disk backup (`monthly_full_bkup_disk` in Appendix A). Note that an older full backup still exists on tape for recovery purposes during this time.

This approach does have inherent risks since there is a period of time every month where a valid backup does not exist in the Flash Recovery Area. Specifically, this is the time when the existing image copy is deleted and replaced with a new backup. In the case of the EDW, there are roughly 7 hours a month where no valid backup exists in the FRA. In theory, the exposure could be decreased by going to quarterly or bi-annual level 0 backups, but this schedule was left in place since the operations group wanted to completely replace the backup copy with some regularity and potentially uncover any unforeseen issues if we were forced to take a full backup in an emergency situation. It is also worth mentioning that if the split mirror backup solution had been chosen, this exposure time would have actually been higher. Assuming there is only one mirror to split, the re-synching of disks must take place on a daily basis. Estimating that the re-synch takes at least 30 minutes, the resynch time works out to 30 min * 30 days = 900 minutes a month. There would still be ~15 hours a month where an "unusable" backup existed if you needed to use the mirror from that time.

Question: What should be backed up to tape and how often? It is not practical to run the 'backup recovery area' command nightly since rolling image copy backups would cause 6 TB to be written to tape every night.

Any datafile copy that gets rolled forward will be picked up by the next `'backup recovery area'` command to tape. Therefore, in our case, a significant portion of the data warehouse would end up getting sent to tape each night with that approach. Therefore, backing up just the *backup sets* in the flash recovery area to tape on a daily basis, which only includes the archived log and incremental backups (not image copies), was determined to be the right solution. The drawback of this approach is that the restore time from tape would increase due to additional incremental backups having to be applied during the recovery process versus having a more current full backup available on tape. The entire flash recovery area, including image copies, are backed up only once a week.

Question: How long does it take to backup 1 TB of data to the FRA?

Unfortunately, this was the first system to use a FRA, so we could not use other systems as a baseline for estimation. Additionally, a UNIX '`dd`' write test would have ideally been performed prior to adding the disks to the FRA, so that the maximum throughput level could be determined. In the initial build, the FRA was comprised of (24) 300 GB 10K drives configured in a RAID 5 (3+1) configuration.

In turn, the FRA disk group was presented with (20) 193 GB MetaVolumes (LUNs). The tests were conducted with 2, 3, 4, 5 and 6 RMAN channels and it was determined that I/O saturation was achieved with 4 channels. It was observed that adding more than 4 channels demonstrated a significant "roll off" in performance as opposed to linear growth, which was exhibited by 2, 3, and 4 channels. Initial tests yielded 500-570 GB/hr overall backup throughput to the FRA. However, in these initial tests, not all available disks were allocated to the FRA disk group. When a second equivalent set of disks and MetaVolumes were added, backup rates of approximately 800 GB/hr were achieved.

An RMAN disk backup that reads and writes terabytes of information across the SAN has a noticeable performance impact. Hence, the incrementally updated image copies are key to the success of the backup strategy since they avoid the high cost of a full backup. After obtaining the required results from the disk-based backup testing, we verified the tape-based backup would also complete in a reasonable length of time.

Question: How long does it take to backup 1 TB of FRA backups to tape?

Tuning the tape backup was a more daunting task than originally anticipated. Since there was no similarly sized system in-house for comparison, a theoretical drive throughput was determined. Knowing the expected throughput will ensure you are not wasting capacity, system resources, and most importantly, company money. In addition to normal means of assessing tape drive throughput as mentioned in various backup and recovery white papers on OTN [14], you can also consider running backups to 2, 3, and 4 drives. If the backup times do not exhibit a linear performance characteristic, then you may have a bottleneck in your system. In our case, we had inherited a NetBackup Media Server and although the network trunking and Media Server parameters were thought to be configured properly, we could not drive more than 100 MB/s from the server. This is significant since the four drives were expected to have an average throughput of 180 MB/s compressed. After some investigation, it was determined that the bus speed on the Media Server was the limiting factor. After swapping out the server with a newer Sun T2000 which had a bus speed 10x faster, the throughput barrier was eliminated, but additional tuning was still necessary. It was found that setting both the TCP Window Size on each cluster node and Net Buffer Size on the Media server to 512K were required in order to attain an acceptable throughput 625-650 GB/hr for 4 LTO2 tape drives. A good deal of knowledge was gained from this backup tuning exercise and the next time more tape drives are added to the system, we will know what to expect and how to correctly tune the overall backup environment.

CONCLUSION

With database sizes currently growing into the terabytes and even petabytes, it becomes all the more critical to ensure that the database backup strategy can meet acceptable RTOs, RPOs, and complete within prescribed backup windows. The key to developing a sound strategy is to methodically work through the following steps:

1. Understand database layout and change characteristics
2. Assess recovery requirements
3. Architect backup hardware and storage, based on performance/cost
4. Evaluate backup tools, leveraging Oracle features as much as possible to minimize deployment costs
5. Develop backup strategy based on recovery requirements and available tools
6. Iteratively tune backup environment, as needed
7. Develop and test recovery strategies

In designing a backup and recovery strategy to accomodate a 10 TB and growing data warehouse that services critical store and sales analysis applications, the Oracle DBA team at Starbucks put these principles in action by:

- Implementing time-based partitioning and read-only tablespaces, to reduce amount of data to be backed up
- Identifying the need for faster point-in-time recovery from disk versus relying on tape
- Evaluating Oracle-managed disk-based backups versus third party split mirror solution and deciding that Oracle solution was less expensive, easier to maintain, and offered advantages such as the Flash Recovery Area, fast incremental backups via block change tracking, and fast recovery time via incrementally updated backups
- Investigating how to minimize disk and tape backup space consumption, while still satisfying recovery requirements
- Tuning the hardware environment so that disk and tape backups fully utilized available disks and network bandwidth, to meet required backup windows

REFERENCES

1. Spotlight on Oracle Key Findings from the 2005 WinterCorp TopTen Program
   http://www.oracle.com/solutions/business_intelligence/docs/spotlight2005_6live.pdf

   2005 WinterCorp TopTen Winners
   http://wintercorp.com/VLDB/2005_TopTen_Survey/2005TopTenWinners.pdf

2. Oracle Recovery Manager
   http://www.oracle.com/technology/deploy/availability/htdocs/rman_overview.htm

3. Oracle Secure Backup Overview
http://www.oracle.com/technology/products/secure-backup/index.html

4. Oracle Flashback Technologies
http://www.oracle.com/technology/deploy/availability/htdocs/Flashback_Overview.htm

5. Oracle Data Guard
http://www.oracle.com/technology/deploy/availability/htdocs/DataGuardOverview.html

6. Metalink Note 302615.1 – RMAN & Split Mirror Disk Backups

7. Metalink Note 604683.1 – Supported Backup, Restore and Recovery Operations using Third Party Snapshot Technologies

8. Using Recovery Manager with Oracle Data Guard in Oracle Database 10*g*
http://www.oracle.com/technology/deploy/availability/pdf/RMAN_Data_Guard_10g_wp.pdf

9. Oracle Database 11g Data Guard Concepts and Administration Guide
http://download.oracle.com/docs/cd/B28359_01/server.111/b28294/rman.htm#CHDHAJFD

10. Oracle Advanced Compression with Oracle Database 11*g* Release 1
http://www.oracle.com/technology/deploy/performance/pdf/aco_11gr1_twp_v2.pdf

11. Oracle Database 11*g* Backup and Recovery User's Guide
http://download.oracle.com/docs/cd/B28359_01/backup.111/b28270/rcmconfb.htm#i1019160

12. Oracle LogMiner
http://www.oracle.com/technology/deploy/availability/htdocs/LogMinerOverview.htm

13. Oracle Database 11*g* Backup and Recovery User's Guide
http://download.oracle.com/docs/cd/B28359_01/backup.111/b28270/rcmbckad.htm#CHDJCAAG

14. Oracle Backup and Recovery Collateral
http://www.oracle.com/technology/deploy/availability/techlisting.html#BR

APPENDIX A - STARBUCKS RMAN SCRIPTS

`Daily_incr_bkup_disk` – Daily Backup to FRA

```
{
 recover device type disk copy of database with
tag 'full_bkup' until time 'sysdate - 2';
```

```
 backup incremental level 1 for recover of copy
with tag full_bkup database;

 backup as backupset archivelog all not backed
up delete all input;
 delete noprompt obsolete recovery window of 2
days device type disk;
}
```

Daily_full_bkup_tape – Daily Backup of Incremental and Archived Log Backups to Tape

```
{
 allocate channel backup_1 DEVICE TYPE
'SBT_TAPE' PARMS
'ENV=(NB_ORA_CLIENT=server.domain.net,NB_ORA_PO
LICY=ora_policy,NB_ORA_SERV=master.domain.net,N
B_ORA_SCHED=Backup)' FORMAT   '%U';

 allocate channel backup_2 DEVICE TYPE
'SBT_TAPE' PARMS
'ENV=(NB_ORA_CLIENT=server.domain.net,NB_ORA_PO
LICY=ora_policy,NB_ORA_SERV=master.domain.net,N
B_ORA_SCHED=Backup)' FORMAT   '%U';

 backup backupset all;
 release channel backup_1;
 release channel backup_2;
}
```

Weekly_full_bkup_tape – Weekly backup of FRA to Tape

```
{
 backup recovery area;
}
```

Monthly_full_bkup_disk – Monthly Full Backup to Disk

```
{
 delete force noprompt datafilecopy tag
full_bkup;
 backup incremental level 1 for recover of copy
with tag full_bkup database;
 backup as backupset archivelog all not backed
up delete all input;
}
```

Very Large Database Backup & Recovery Best Practices
July 2008
Authors: Tim Chien (Oracle), Greg Green (Starbucks)

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com