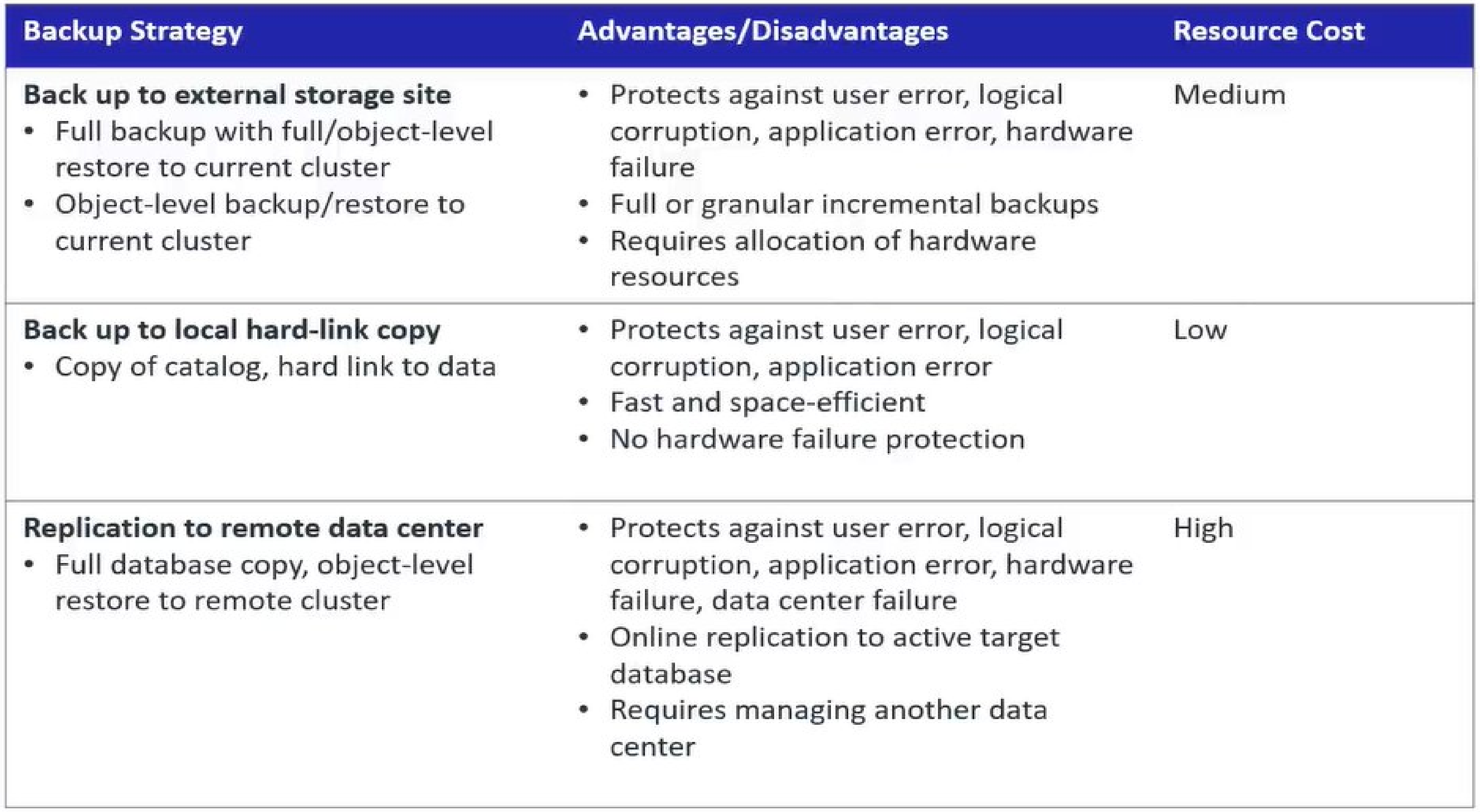
VERTICA
by opentext™

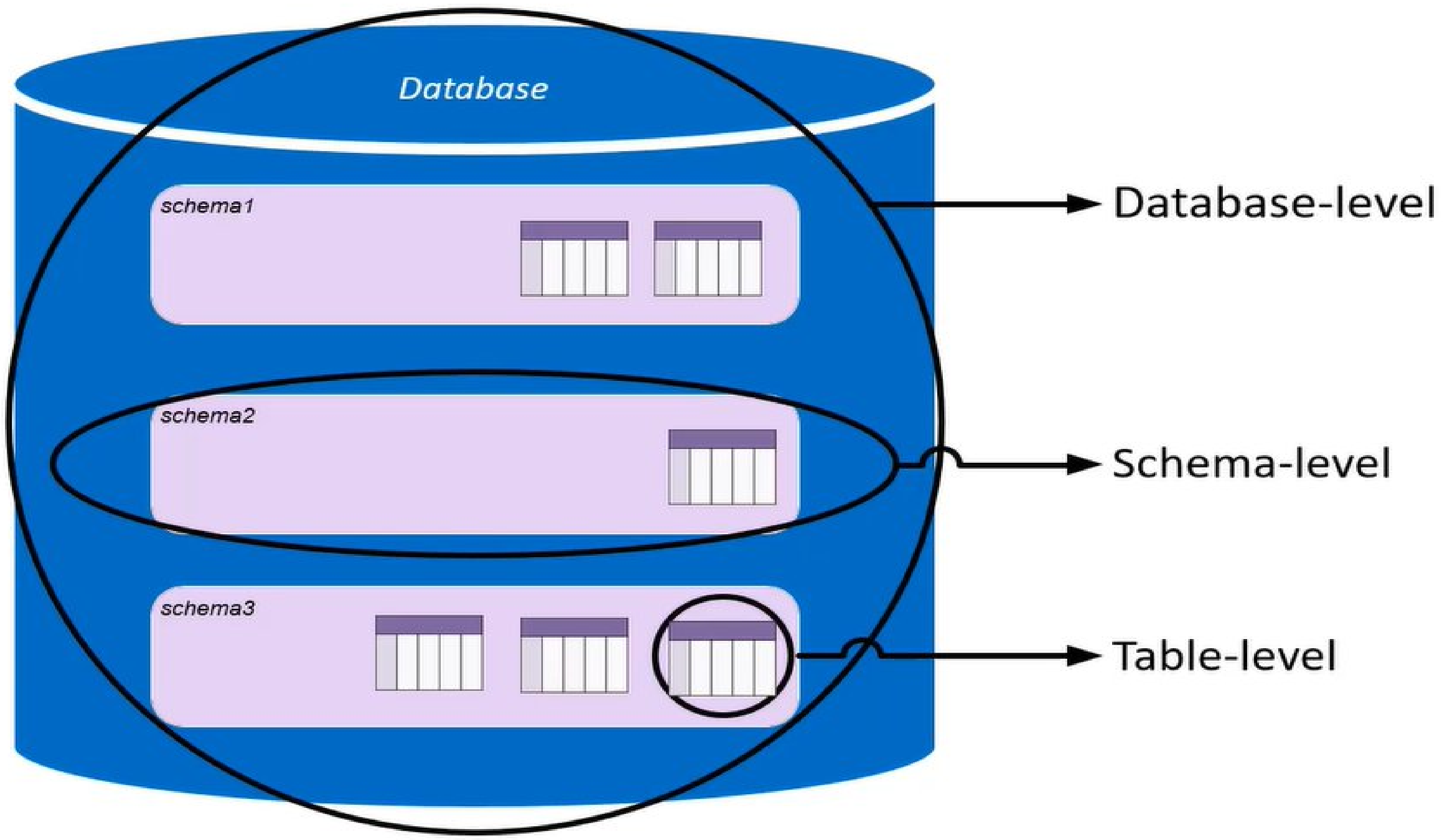
# Backup and Restore

# Levels of Data Protection



| Backup Strategy | Advantages/Disadvantages | Resource Cost |
|---|---|---|
| **Back up to external storage site**<br>• Full backup with full/object-level restore to current cluster<br>• Object-level backup/restore to current cluster | • Protects against user error, logical corruption, application error, hardware failure<br>• Full or granular incremental backups<br>• Requires allocation of hardware resources | Medium |
| **Back up to local hard-link copy**<br>• Copy of catalog, hard link to data | • Protects against user error, logical corruption, application error<br>• Fast and space-efficient<br>• No hardware failure protection | Low |
| **Replication to remote data center**<br>• Full database copy, object-level restore to remote cluster | • Protects against user error, logical corruption, application error, hardware failure, data center failure<br>• Online replication to active target database<br>• Requires managing another data center | High |

# Database Backup Levels



## Database Backup Levels

- On a regular schedule, as part of regular database maintenance
- Before and after upgrading Vertica
- Before and after a single load of a large volume of data
- Before dropping partitions
- Before adding, removing, or replacing nodes
- After recovering a cluster from a crash

## Backup – Overview

- Vertica data files are write-once
- Number of files increases with each load
- Tuple Mover keeps the number of files under control
- To backup, copy Vertica files to stable storage

## Backup and Restore Options

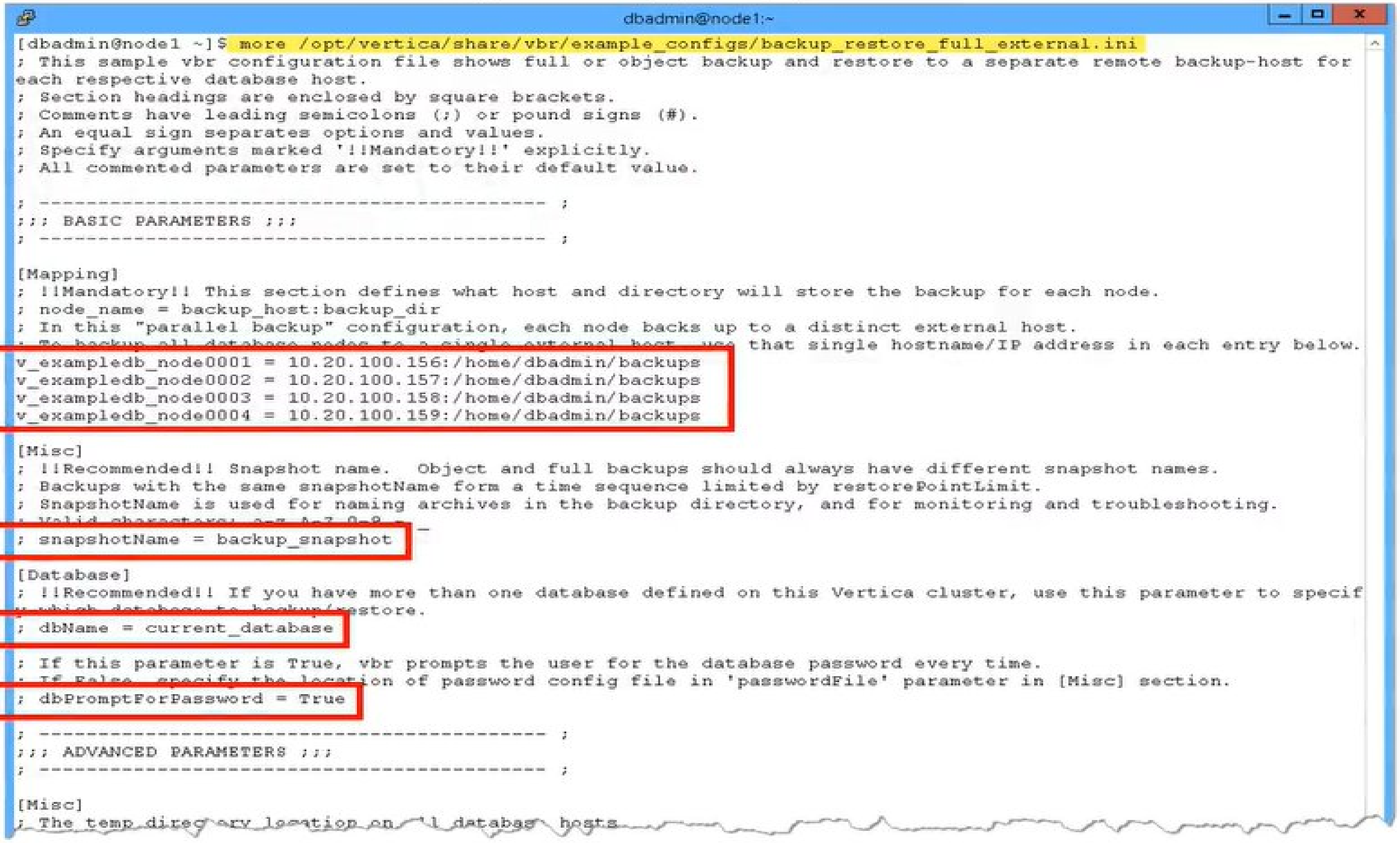
- Backup and Restore by Database
  - General backup process
- Backup and Restore by Schema
  - Multi-tenant database with different backup frequency
  - Backup and Restore by Table
  - Backup data specific to Vertica
  - Verify space, restore single table, etc.

## Creating Backup Configuration File

- Located at /opt/vertica/bin/vbr.py
- Create configuration file first
  - vbr.py - -setupconfig
  - Defines where the database backup is saved, the temporary directories to use, and which nodes, schema(s), and/or table(s) in the database are to be backed up

## Sample Backup Configuration File

backup_restore_full_external.ini



```
[dbadmin@node1 ~]$ more /opt/vertica/share/vbr/example_configs/backup_restore_full_external.ini
; This sample vbr configuration file shows full or object backup and restore to a separate remote backup-host for
each respective database host.
; Section headings are enclosed by square brackets.
; Comments have leading semicolons (;) or pound signs (#).
; An equal sign separates options and values.
; Specify arguments marked '!!Mandatory!!' explicitly.
; All commented parameters are set to their default value.

; ------------------------------------------- ;
;;; BASIC PARAMETERS ;;;
; ------------------------------------------- ;

[Mapping]
; !!Mandatory!! This section defines what host and directory will store the backup for each node.
; node_name = backup_host:backup_dir
; In this "parallel backup" configuration, each node backs up to a distinct external host.
; To backup all database nodes to a single external host, use that single hostname/IP address in each entry below.
v_exampledb_node0001 = 10.20.100.156:/home/dbadmin/backups
v_exampledb_node0002 = 10.20.100.157:/home/dbadmin/backups
v_exampledb_node0003 = 10.20.100.158:/home/dbadmin/backups
v_exampledb_node0004 = 10.20.100.159:/home/dbadmin/backups

[Misc]
; !!Recommended!! Snapshot name.  Object and full backups should always have different snapshot names.
; Backups with the same snapshotName form a time sequence limited by restorePointLimit.
; SnapshotName is used for naming archives in the backup directory, and for monitoring and troubleshooting.
; Valid characters: a-z A-Z 0-9 - _
; snapshotName = backup_snapshot

[Database]
; !!Recommended!! If you have more than one database defined on this Vertica cluster, use this parameter to specif
y which database to backup/restore.
; dbName = current_database

; If this parameter is True, vbr prompts the user for the database password every time.
; If False, specify the location of password config file in 'passwordFile' parameter in [Misc] section.
; dbPromptForPassword = True

; ------------------------------------------- ;
;;; ADVANCED PARAMETERS ;;;
; ------------------------------------------- ;

[Misc]
; The temp directory location on all database hosts.
```

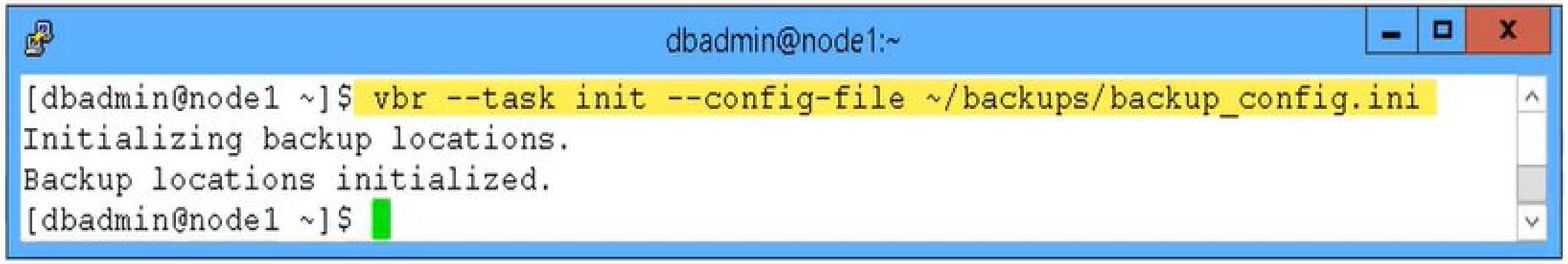
## Backup Preperation (1 of 2)

- Verify your database is running
- All of the backup hosts must be up and available
- The location to store the backups must have  sufficient disk space to store the backups, and must  be writable by the user account used to start the  backup utility

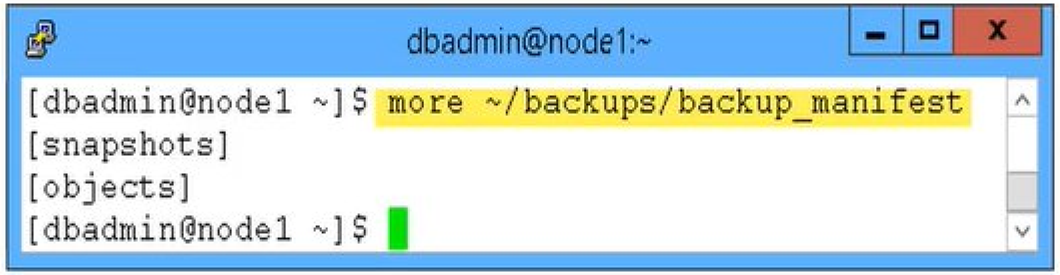
## Backup Preparation (2 of 2)

- Backups are stored in the location you specify in the  configuration file you use to back up the database
- The directory containing the backup file has a  subdirectory for each node backed up to that  location, which in turn contains a directory with the  name of the backup snapshot
  - The snapshot name is set using the snapshotName option in the configuration file

## Initializing the Backup Location



```
[dbadmin@node1 ~]$ vbr --task init --config-file ~/backups/backup_config.ini
Initializing backup locations.
Backup locations initialized.
[dbadmin@node1 ~]$
```

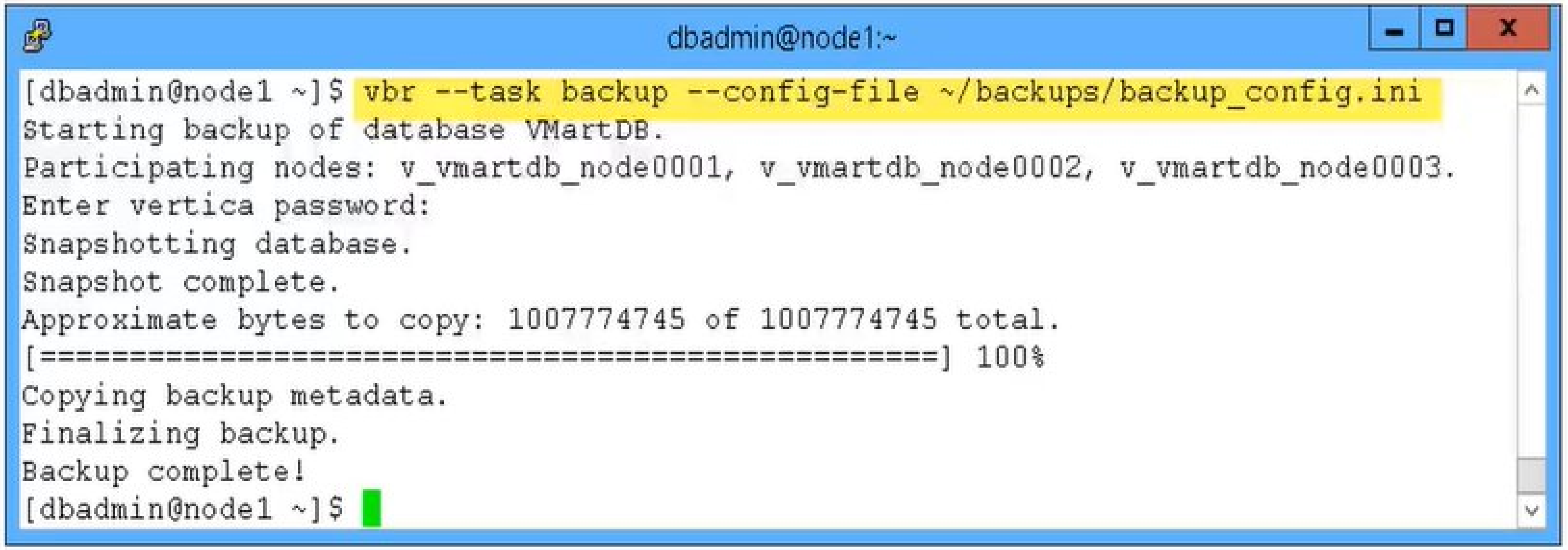


```
[dbadmin@node1 ~]$ more ~/backups/backup_manifest
[snapshots]
[objects]
[dbadmin@node1 ~]$
```

## Performing a Backup

- vbr.py - -task backup - -config-file <configfile>
  used for full and incremental backups
- First run does a full backup
- Subsequent runs only copy files added since the last  backup
  - Vertica’s files are write-once
  - Files are only added or deleted, never modified

## Running the Backup

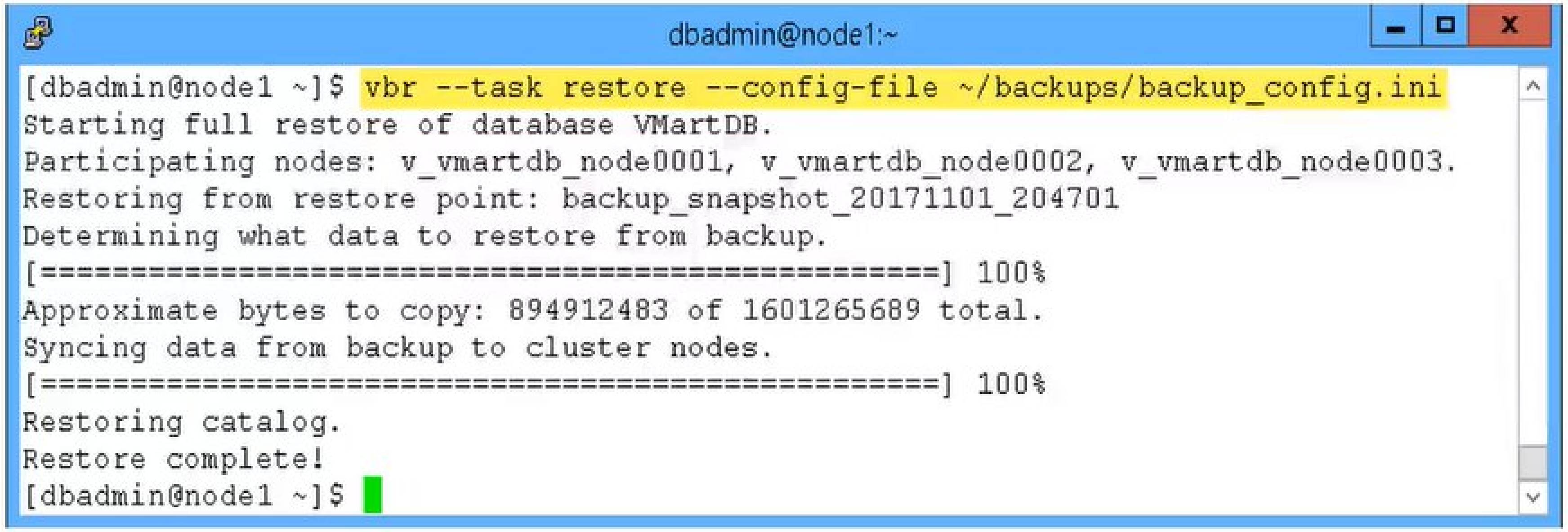


```
[dbadmin@node1 ~]$ vbr --task backup --config-file ~/backups/backup_config.ini
Starting backup of database VMartDB.
Participating nodes: v_vmartdb_node0001, v_vmartdb_node0002, v_vmartdb_node0003.
Enter vertica password:
Snapshotting database.
Snapshot complete.
Approximate bytes to copy: 1007774745 of 1007774745 total.
[==================================================] 100%
Copying backup metadata.
Finalizing backup.
Backup complete!
[dbadmin@node1 ~]$
```

## Performing a Restore

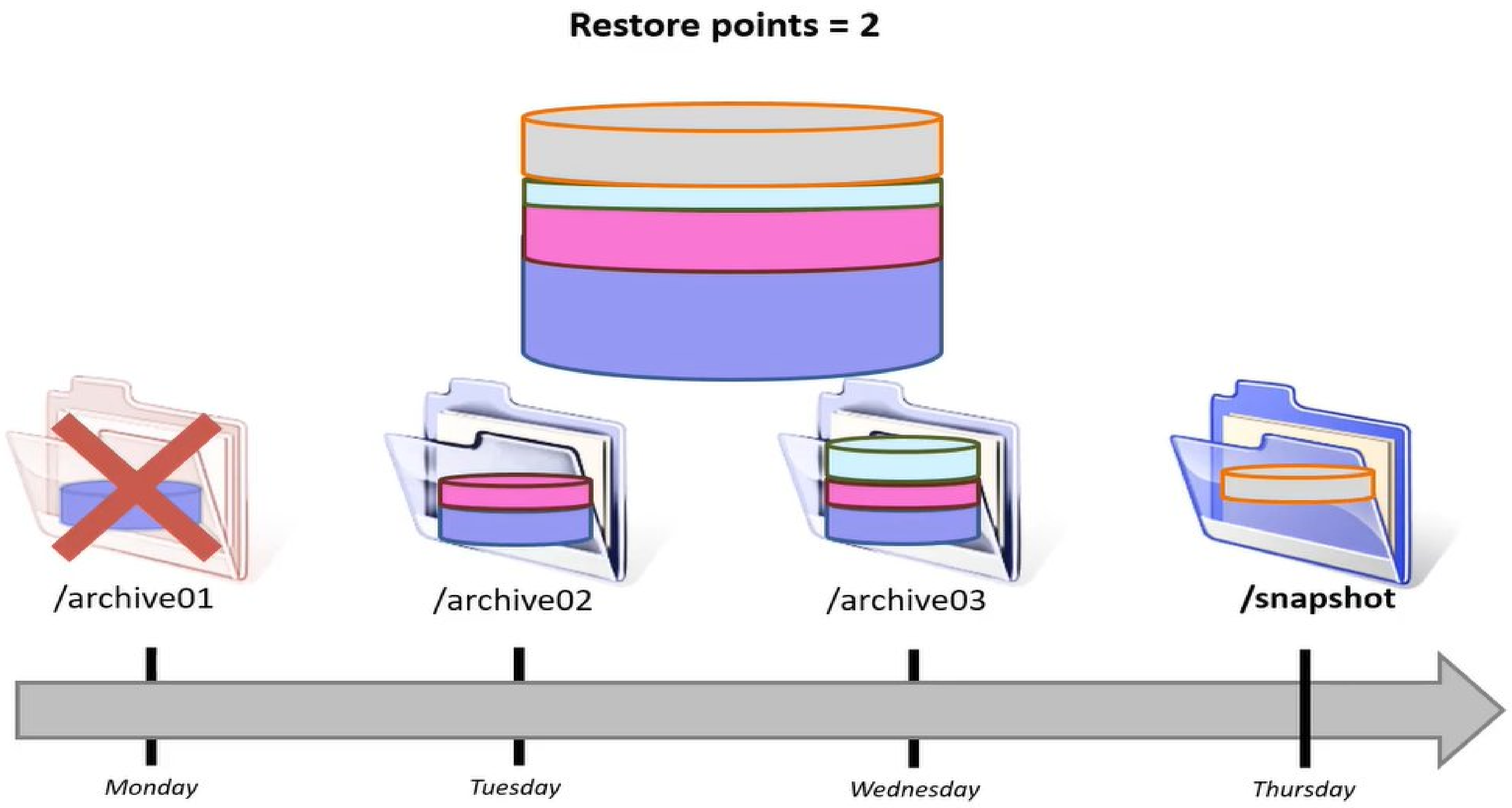
- vbr.py - -task restore - -config-file <configfile>
- Interactive prompts guide the restore process
- Can restore entire database, schema, or table

# Restoring the Database from a Backup

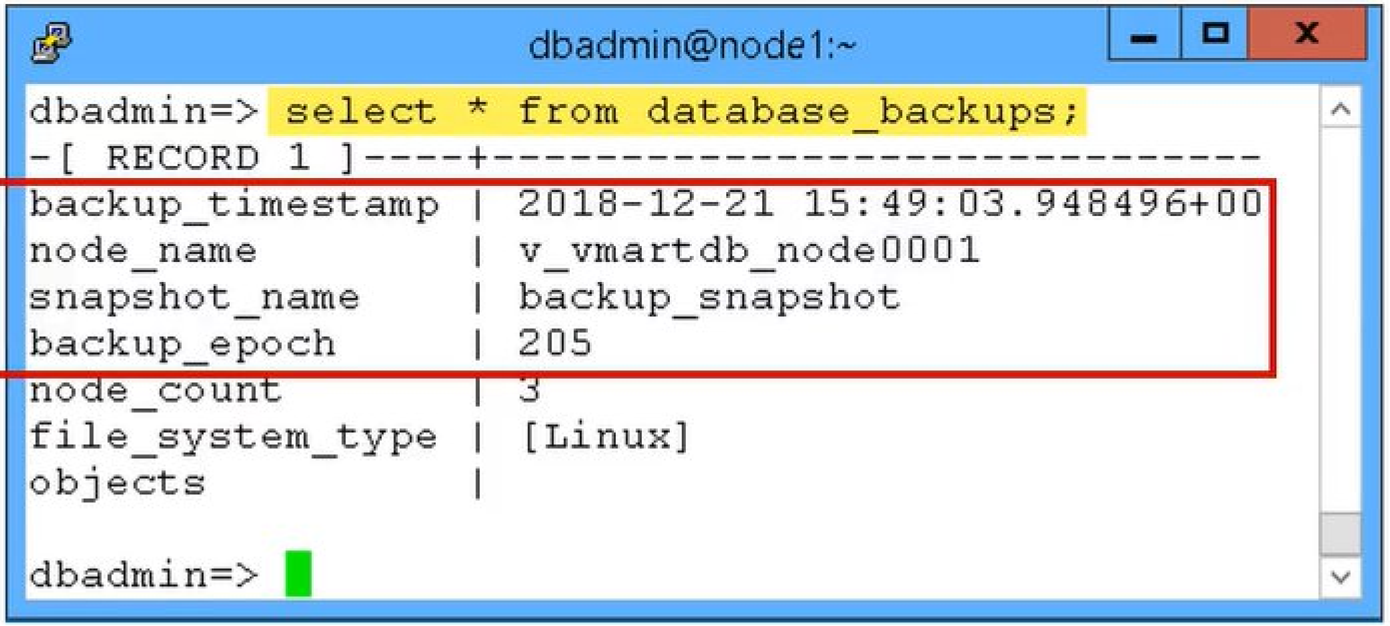


```
[dbadmin@node1 ~]$ vbr --task restore --config-file ~/backups/backup_config.ini
Starting full restore of database VMartDB.
Participating nodes: v_vmartdb_node0001, v_vmartdb_node0002, v_vmartdb_node0003.
Restoring from restore point: backup_snapshot_20171101_204701
Determining what data to restore from backup.
[==================================================] 100%
Approximate bytes to copy: 894912483 of 1601265689 total.
Syncing data from backup to cluster nodes.
[==================================================] 100%
Restoring catalog.
Restore complete!
[dbadmin@node1 ~]$
```

## Backup Restore points



## Monitoring Backups



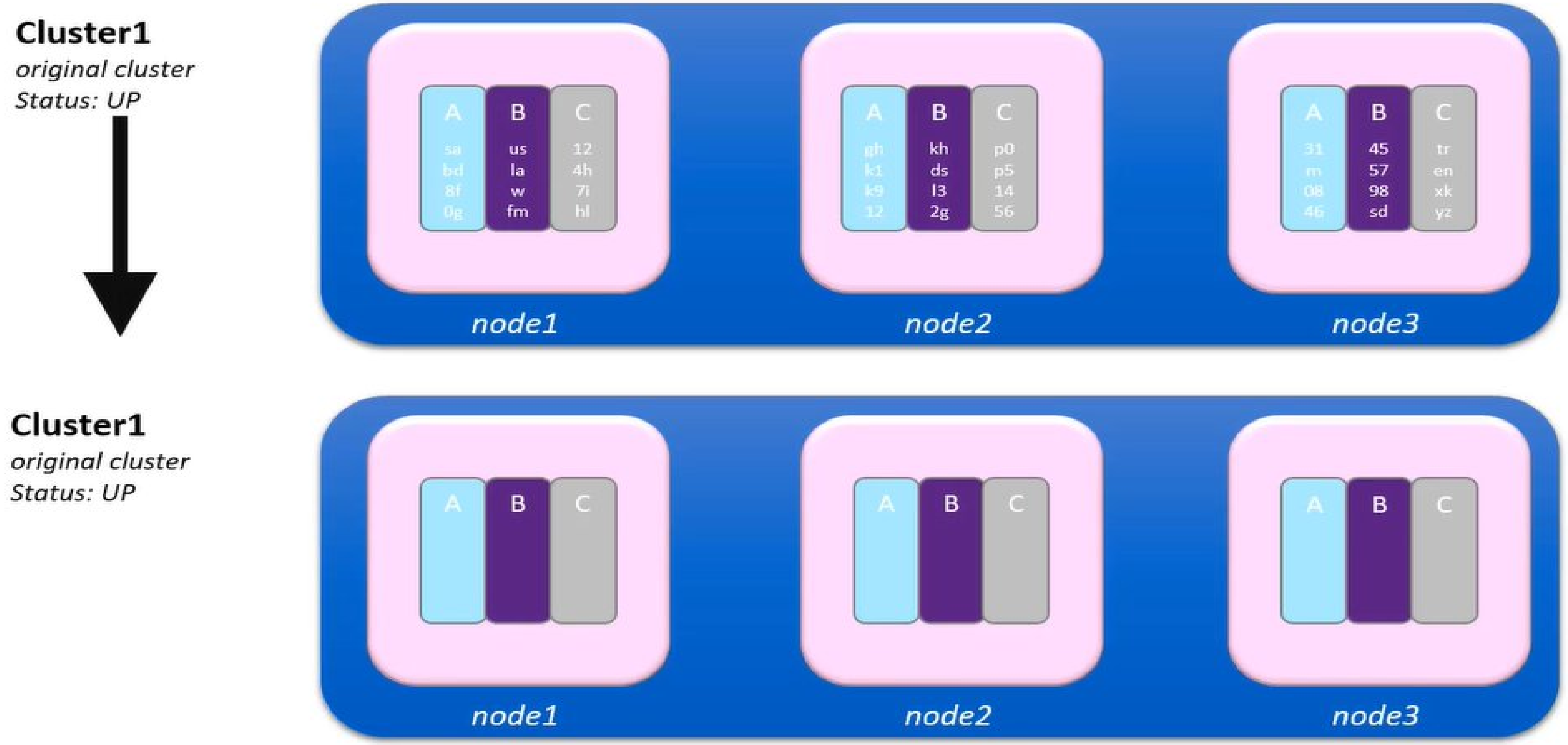
dbadmin@node1:~

```
dbadmin=> select * from database_backups;
-[ RECORD 1 ]----+----------------------------
backup_timestamp | 2018-12-21 15:49:03.948496+00
node_name        | v_vmartdb_node0001
snapshot_name    | backup_snapshot
backup_epoch     | 205
node_count       | 3
file_system_type | [Linux]
objects          |

dbadmin=>
```

# Restoring Objects

vbr --task **restore** --config-file *filename.ini* --**restore-objects** *obj1,..objN*



## Copy Vertica database

- Target cluster requirements:
  - Same number of nodes source cluster
  - Database with the same name as the database being copied
  - Same node names as the source cluster
  - Same database administrator account
- vbr.py - -task copycluster - -config-file <configfile>
- Ideal for creating a dormant Disaster Recovery site

## Node Recovery (1 of 2)

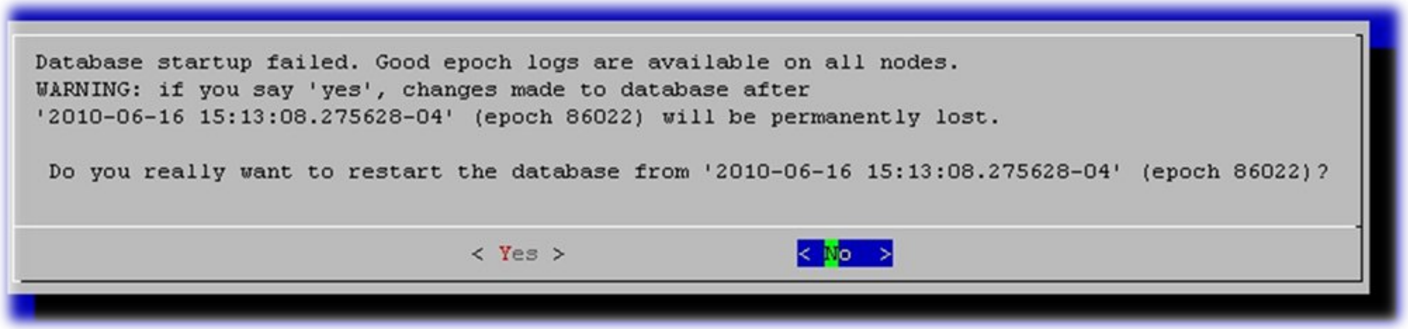
- A node can rebuild its data set from other nodes in  the cluster if the cluster is K-safe
- Full recovery
  - – Node rebuilds from scratch

## Node Recovery (2 of 2)

- Incremental recovery
  - Node rebuilds from current persisted state
  - To speed up a full recovery, use a prior backup for the given node and perform incremental recovery
- RAID arrays (5,6,10) can be rebuilt without impact to other cluster nodes

# Recover from Last Good Epoch (LGE)

If nodes contain persistent data from different epochs, the Last Good  Epoch (LGE) on all nodes is used to determine recovery point



```
Database startup failed. Good epoch logs are available on all nodes.
WARNING: if you say 'yes', changes made to database after
'2010-06-16 15:13:08.275628-04' (epoch 86022) will be permanently lost.

 Do you really want to restart the database from '2010-06-16 15:13:08.275628-04' (epoch 86022)?

                    < Yes >                    < No  >
```

```
*** Restarting database db at epoch 86022 ***

Node Status: v_db_node0001: (DOWN) v_db_node0002: (DOWN) v_db_node0003: (DOWN)

    Node Status: v_db_node0001: (INITIALIZING) v_db_node0002: (INITIALIZING) v_db_node0003: (INITIALIZING)

    Node Status: v_db_node0001: (RECOVERING) v_db_node0002: (RECOVERING) v_db_node0003: (RECOVERING)

    Node Status: v_db_node0001: (UP) v_db_node0002: (UP) v_db_node0003: (UP)
```

opentext™

## Monitoring Recovery

- Monitor disk space
  - df –h
  - SELECT * FROM v_monitor.disk_storage;

- Monitor recovery

  – tail –f <catalog-directory-  path>/vertica.log opentext™

Thank you