SQLBits

Session: Saturday, 14:50

Building a Better Backup Strategy (L200)

Paul S. Randal
paul@SQLskills.com

Paul S. Randal

- **Consultant/Trainer/Speaker/Author**
- **CEO, SQLskills.com**
  - Email: Paul@SQLskills.com
  - Blog: https://www.SQLskills.com/blogs/Paul
  - Twitter: @PaulRandal
  - 5 years at DEC responsible for the VMS file-system and chkdsk
  - Almost 9 years as developer/manager in the SQL Storage Engine team through August 2007, ultimately responsible for Core Storage Engine
- **Instructor-led training, consulting (anything you need)**
- **Online training: http://pluralsight.com/**
- **Get our newsletter: https://www.sqlskills.com/Insider**

How to Plan a Backup Strategy?

Don’t Do It!

Plan A *Restore* Strategy

- I like to say never to plan a backup strategy
- Always plan out what restore operations you want to be able to do, and that will dictate what backups you need and what recovery model to use
- This usually turns out to be full recovery model, with a full + differential + transaction log backup strategy

- Doing it the other way around can lead to not having the right backups to be able to do the restores you need

Overview

- Planning the strategy
- Backup types
- Restore considerations
- Sample backup strategies

Planning the Strategy

- **Even with the most sophisticated redundancy, recovery from total loss of all data centers can only be done using backups**
  - And only then if they're stored somewhere else!
- **Requirements gathering process feeds into the backup/restore strategy**
  - And the fact that storage space and management are required for backups then feeds back into the requirements
- **What restores you need to be able to do depends on (at least):**
  - What you're protecting
  - Downtime SLA (Service Level Agreement)
  - Data loss SLA

Downtime SLA

- **Maximum allowable downtime or RTO (Recovery Time Objective)**
- **Commonly discussed in terms of ‘number of nines’**
  - 5-nines = 99.999% uptime (slightly over 5 minutes downtime per year)
  - 4-nines = 99.99% uptime (almost 52.5 minutes downtime per year)
  - 3-nines = 99.9% uptime (almost 8.75 hours downtime per year)
  - 2-nines = 99% uptime (just over 3.5 days of downtime per year)
- **Must consider how downtime is defined for you**
  - Is it a proportion of 24x7 or, say, 9am-5pm weekdays
- **Lowest RTO means using differential and log backups**
- **5-nines of 24x7 is very hard and requires synchronous HA**
  - Management may well ask for zero downtime!

Data Loss SLA

- **Maximum allowable data loss or RPO (Recovery Point Objective)**
- **Must consider how data loss is defined for you**
  - Usually work done over a period of time
- **May be different for different tables or databases**
- **Zero data-loss is much more easily achievable than 5-nines uptime**
  - Management will likely ask for zero data-loss
- **Zero data-loss means taking log backups in the full recovery model**
  - This may mean *lots* of log backups if the log generation rate is very high and you want to stop the log from having to grow because it hasn’t been backed up

Why are the SLAs Important?

- **RTO limits how long you have to:**
  - Find all the necessary backups
  - Perform all the necessary restores
  - Check that everything’s working again
  - *OR* just failover to a redundant site and not use backups
- **RPO defines how close you have to get to current**
- **These may be different for different portions of your data**
- **These two limitations may mean that you must implement some HA technologies (e.g. SAN replication, database mirroring, availability groups) and rely on restoring from backups as a last resort**

What Are You Protecting?

- **What you’re trying to protect dictates physical layout and sometimes backup type**
- **For example:**
    - Protecting an entire database
        - Sounds simple – but not really
        - Consider size, transaction log generation rate, regulatory compliance
    - Protecting the most recent month’s data in a very large table
        - Requires partitioning, multiple-filegroups, and log backups

Full Backup

Creates image for starting restore sequence

Can be database, filegroup, or file

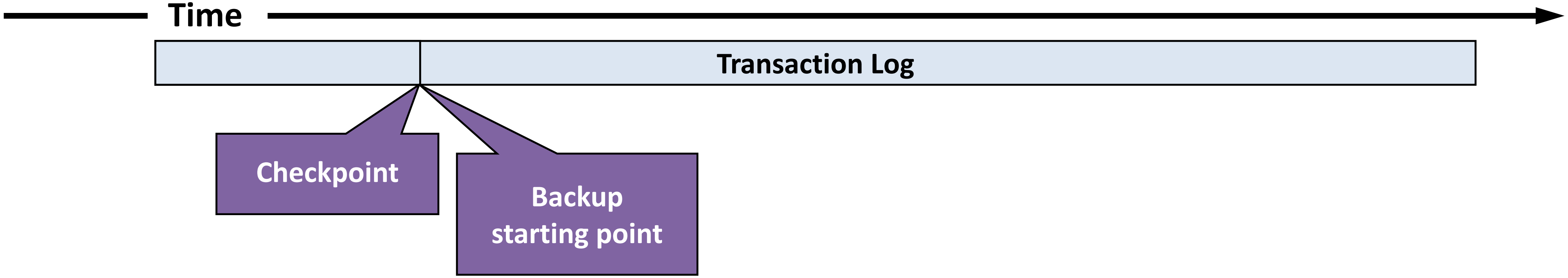
Complete image of all data

Does not cause blocking

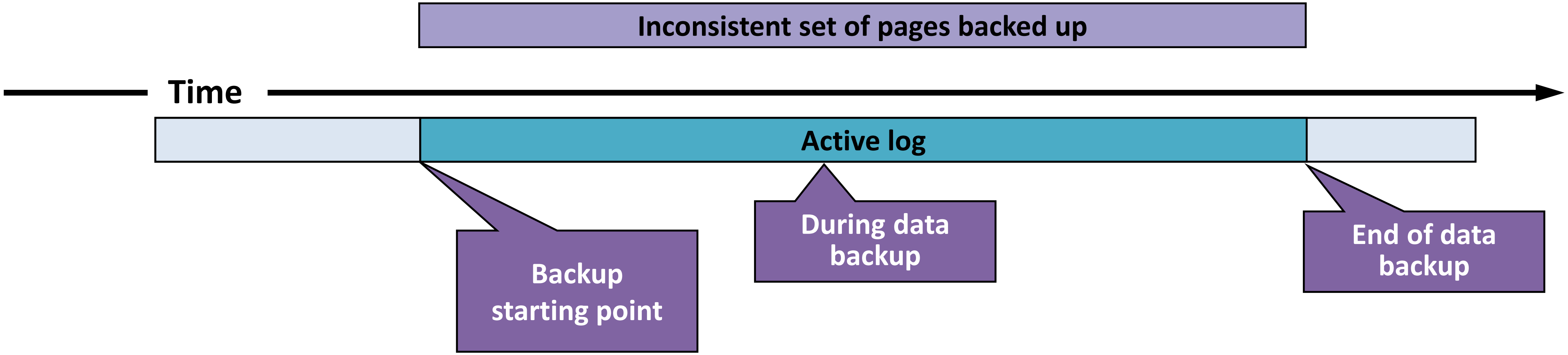
Usually not used alone during restore sequence

Image is of time of backup completion

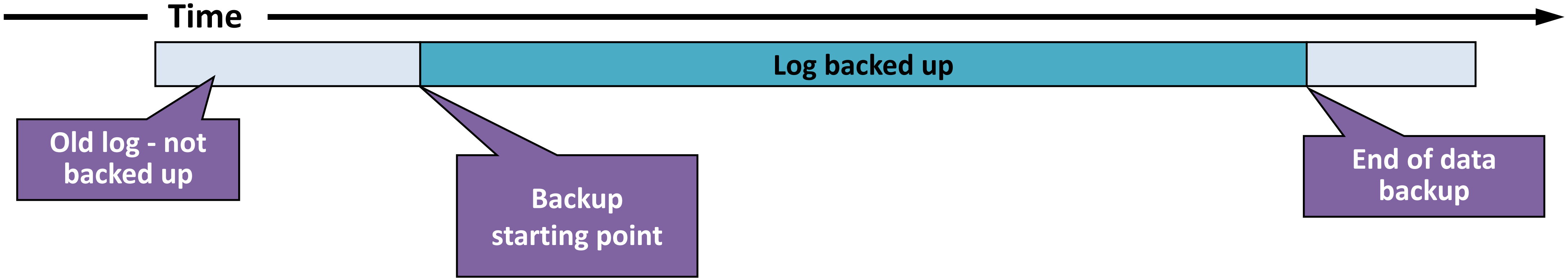
How a Full Database Backup Works



How a Full Database Backup Works



How a Full Database Backup Works



How a Full Database Backup Works

Inconsistent set of pages backed up

+

Log backed up

=

Transactionally-consistent database image

Differential Backup

Creates image for continuing restore sequence

Can be database, filegroup, or file

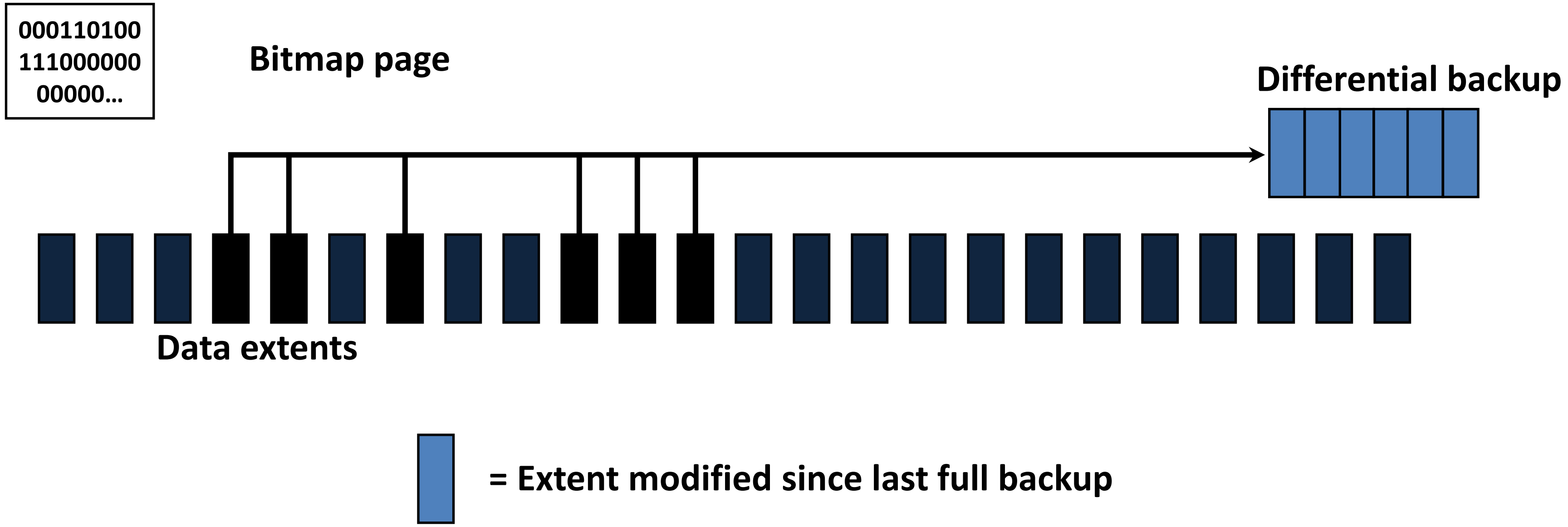
Only data that has changed since most recent full

Does not cause blocking

Only restorable after a full backup restored first

Image is at time of backup completion

How a Differential Backup Works



Transaction Log Backup

Creates image for continuing a restore sequence

Database-level only and allows log clearing

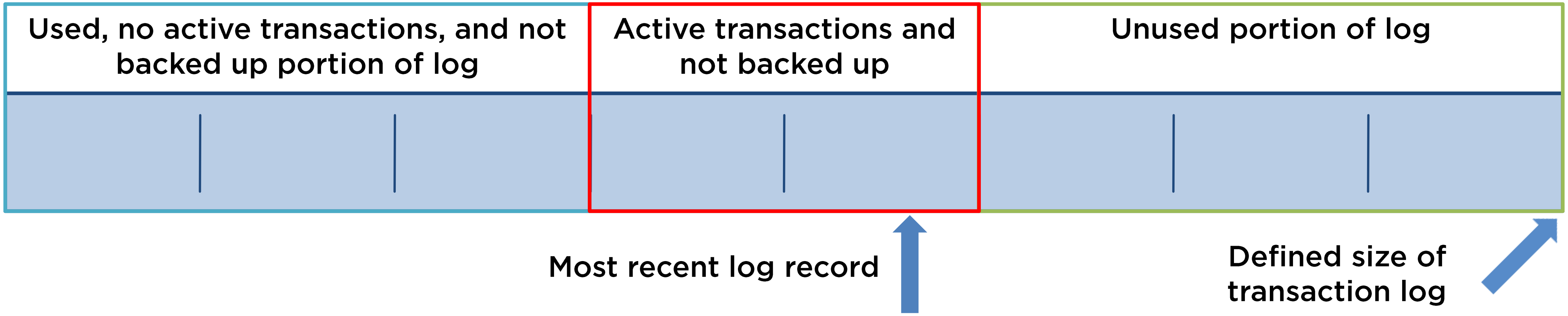
All log records since most recent log backup

Does not cause blocking (except bulk operations)

Only restorable after a full backup restored first

Image is of time of backup completion

How a Log Backup Works



Used, no active transactions, and not backed up portion of log

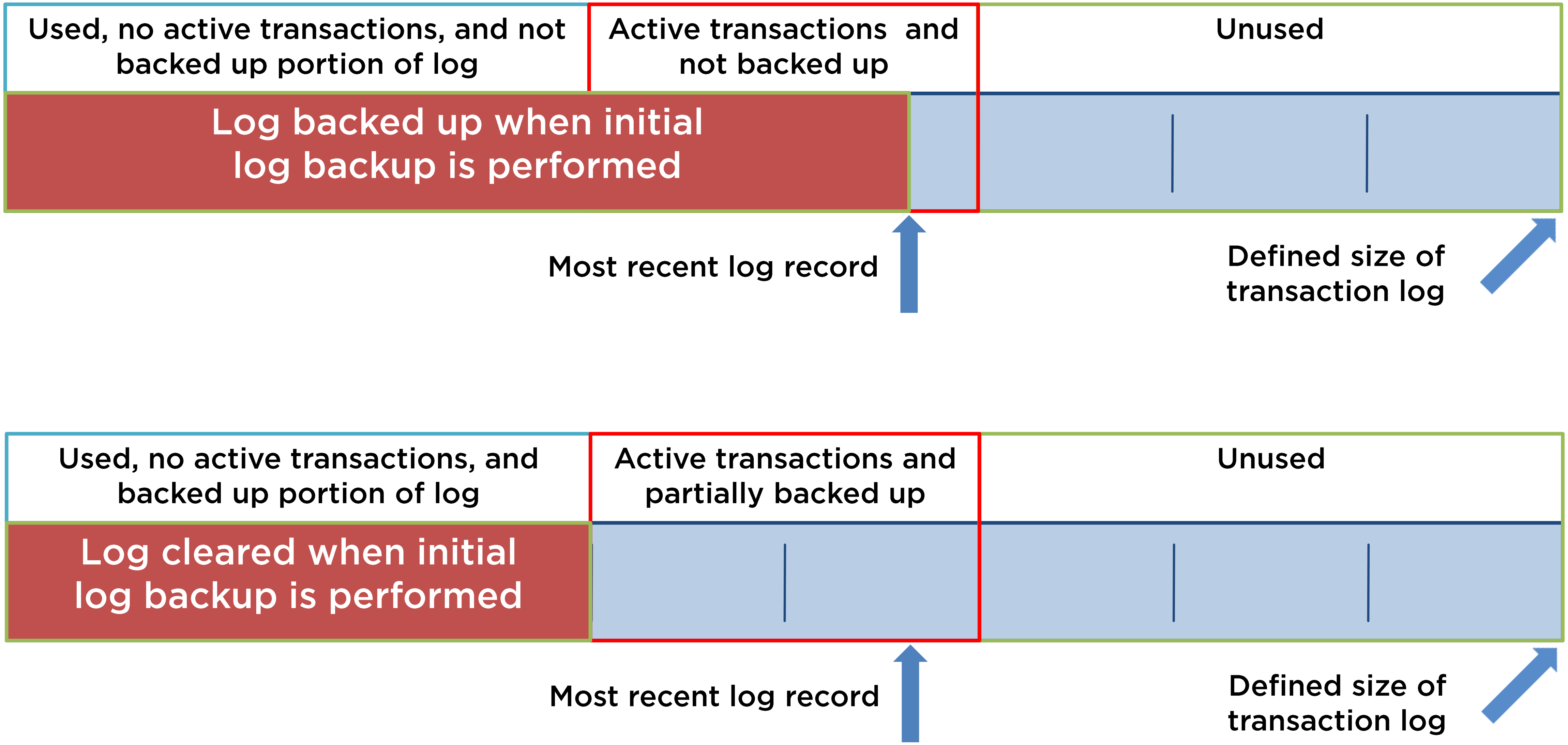
Active transactions and not backed up

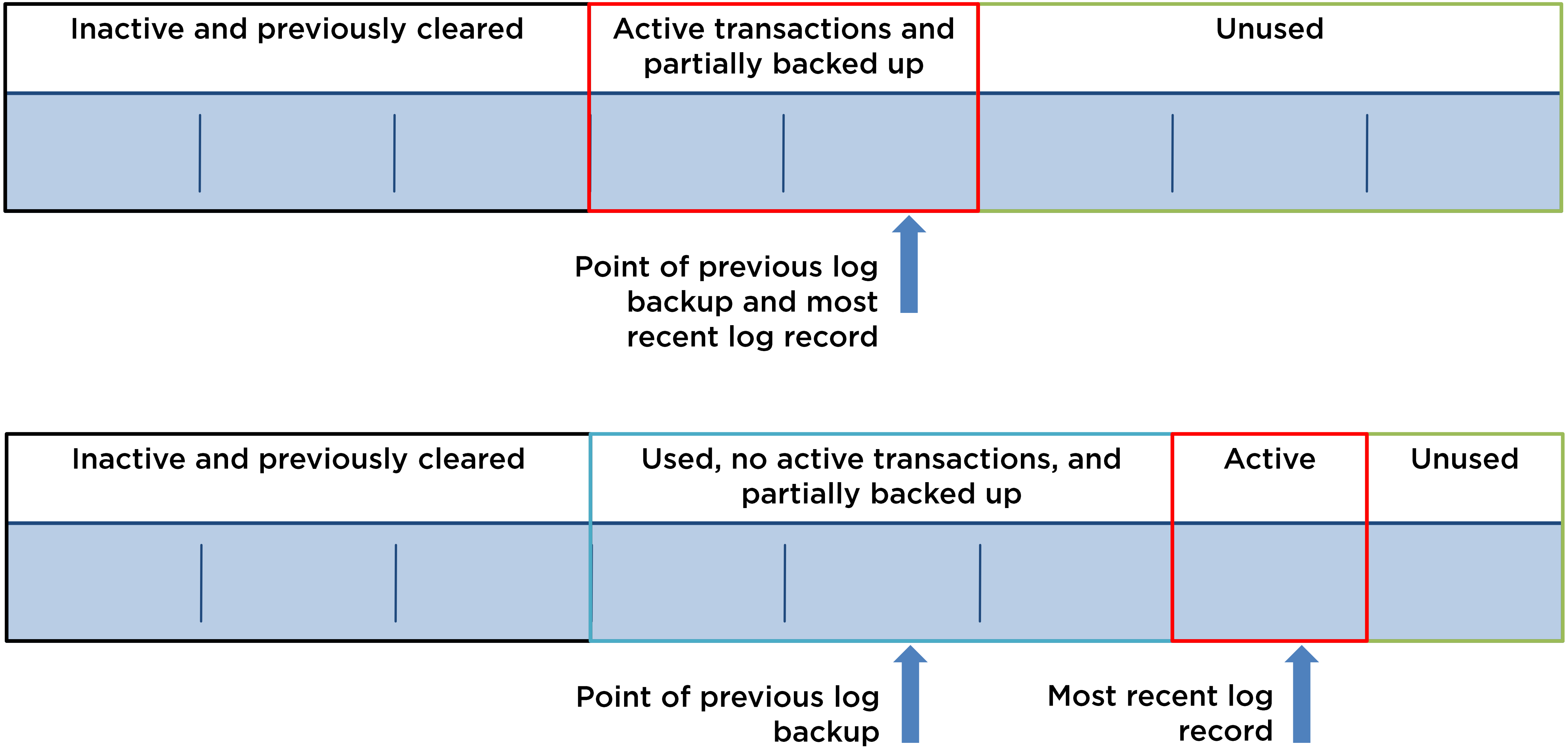
Unused portion of log

Most recent log record

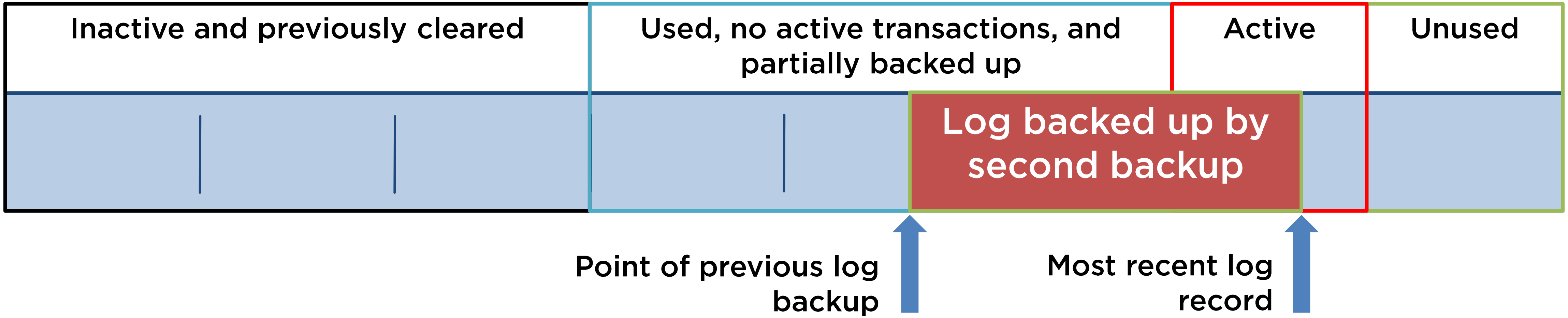
Defined size of transaction log

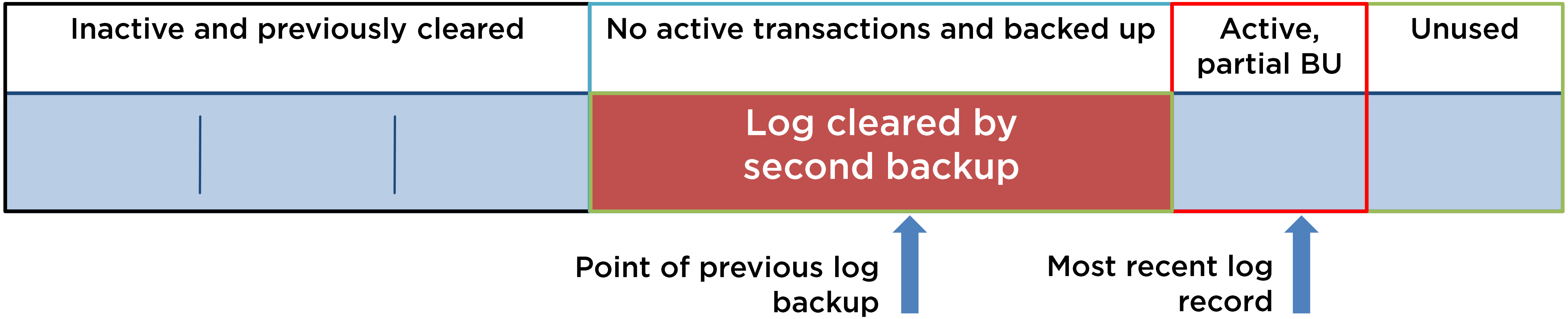
How a Log Backup Works


Used, no active transactions, and not backed up portion of log
Active transactions and not backed up
Unused
Log backed up when initial log backup is performed
Most recent log record
Defined size of transaction log
Used, no active transactions, and backed up portion of log
Active transactions and partially backed up
Unused
Log cleared when initial log backup is performed
Most recent log record
Defined size of transaction log

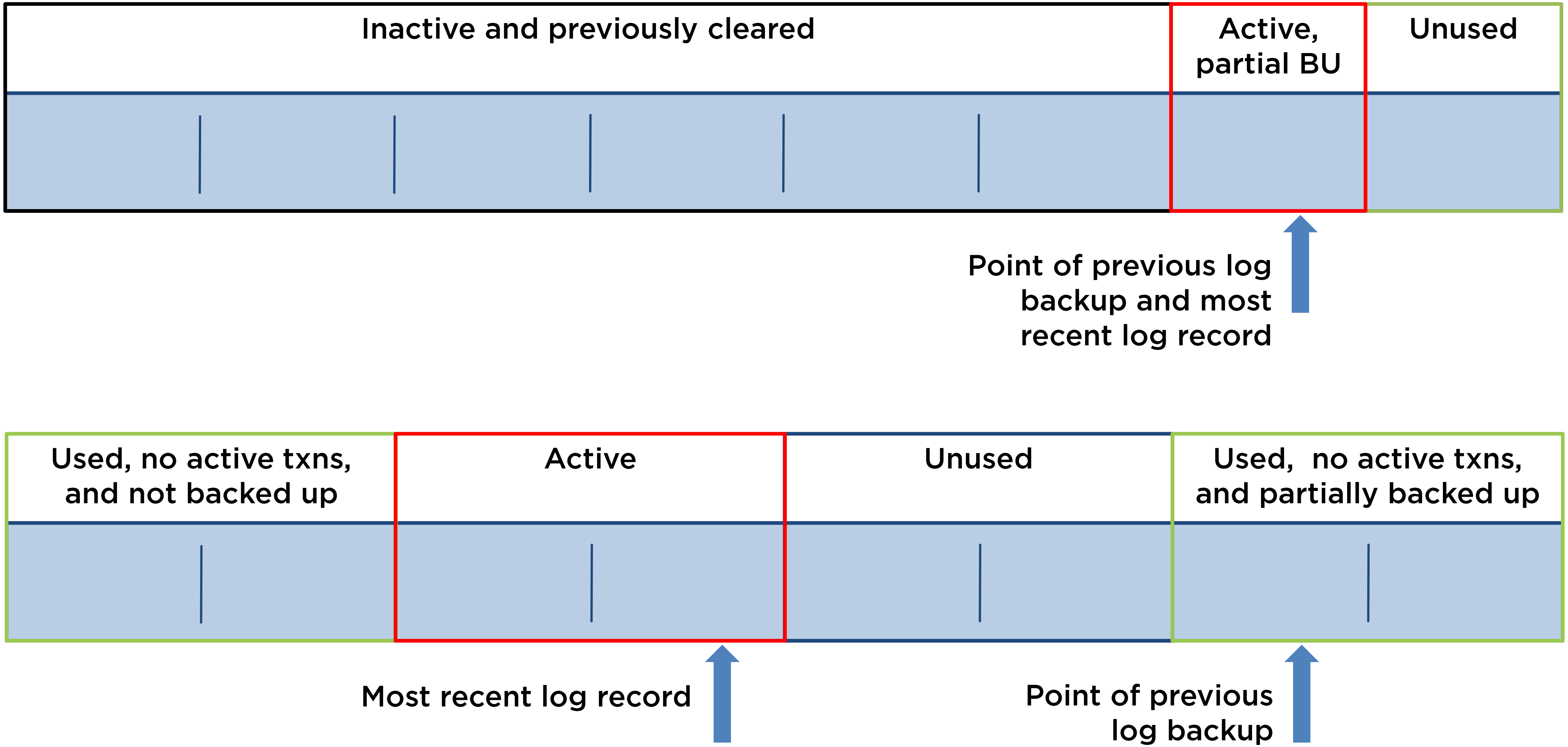

Inactive and previously cleared
Active transactions and partially backed up
Unused
Point of previous log backup and most recent log record
Inactive and previously cleared
Used, no active transactions, and partially backed up
Active
Unused
Point of previous log backup
Most recent log record

After the First Log Backup

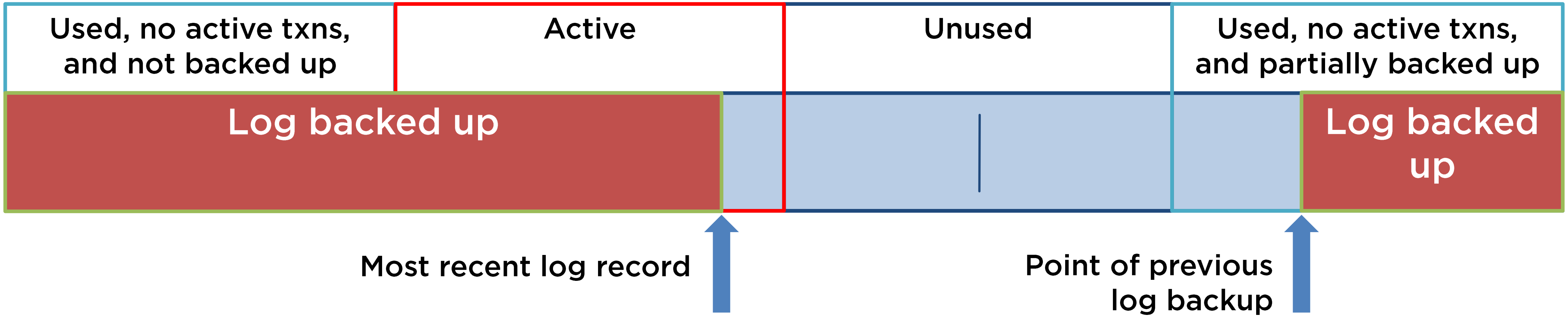

Inactive and previously cleared
Used, no active transactions, and partially backed up
Active
Unused
Log backed up by second backup
Point of previous log backup
Most recent log record


Inactive and previously cleared
No active transactions and backed up
Active, partial BU
Unused
Log cleared by second backup
Point of previous log backup
Most recent log record

Second Log Backup


Inactive and previously cleared
Active, partial BU
Unused
Point of previous log backup and most recent log record
Used, no active txns, and not backed up
Active
Unused
Used, no active txns, and partially backed up
Most recent log record
Point of previous log backup

After the Second Log Backup



Used, no active txns, and not backed up

Active

Unused

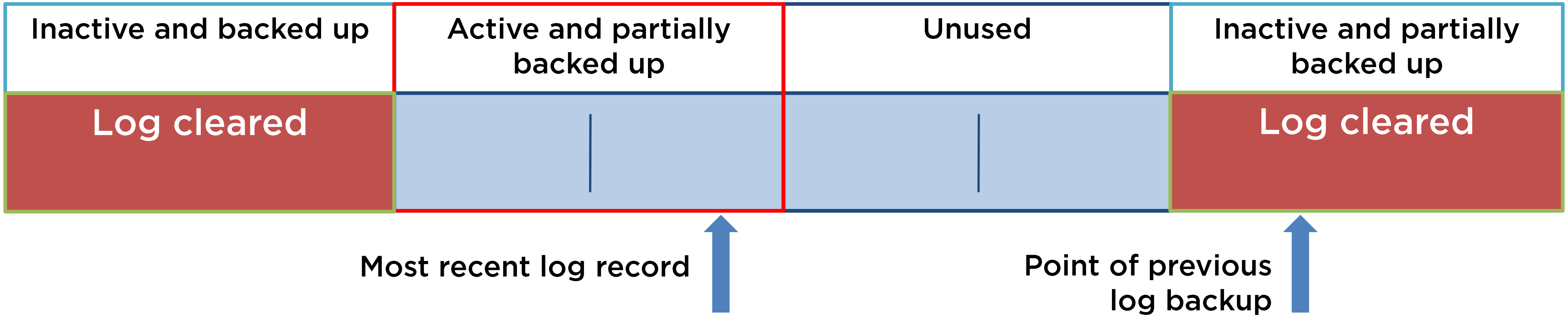
Used, no active txns, and partially backed up

Log backed up

Log backed up

Most recent log record

Point of previous log backup



Inactive and backed up

Active and partially backed up

Unused

Inactive and partially backed up

Log cleared

Log cleared

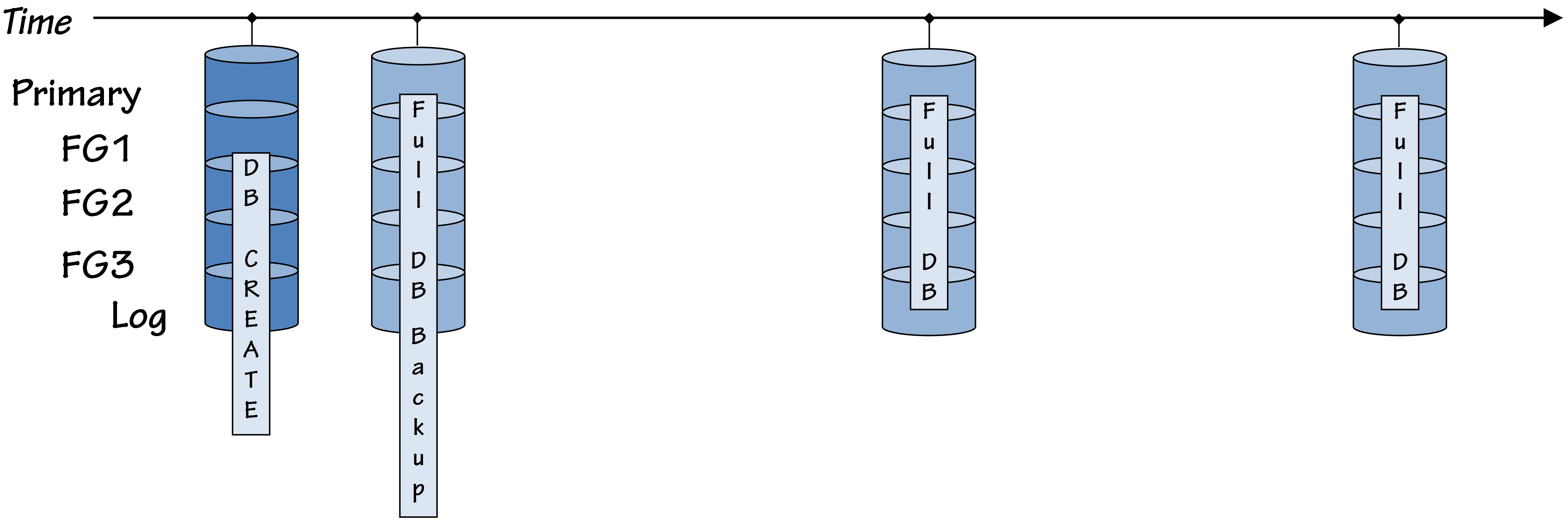
Most recent log record

Point of previous log backup

Third Log Backup

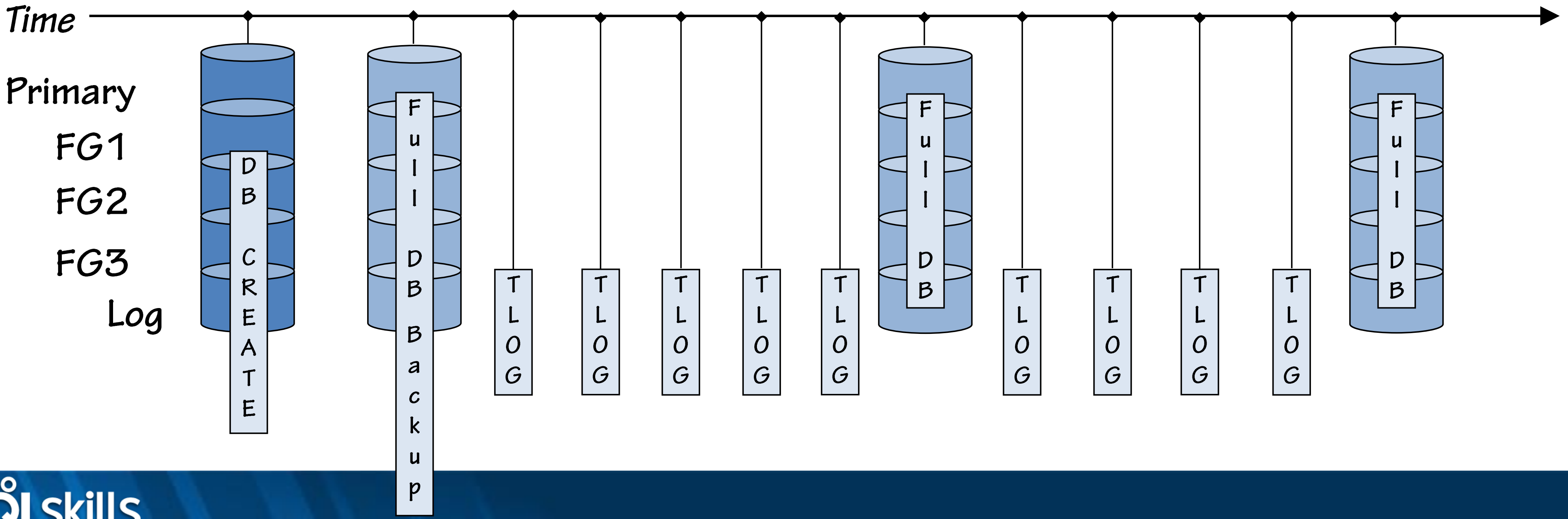
Full Database Backup Only Strategy

- All work since the last full backup is lost



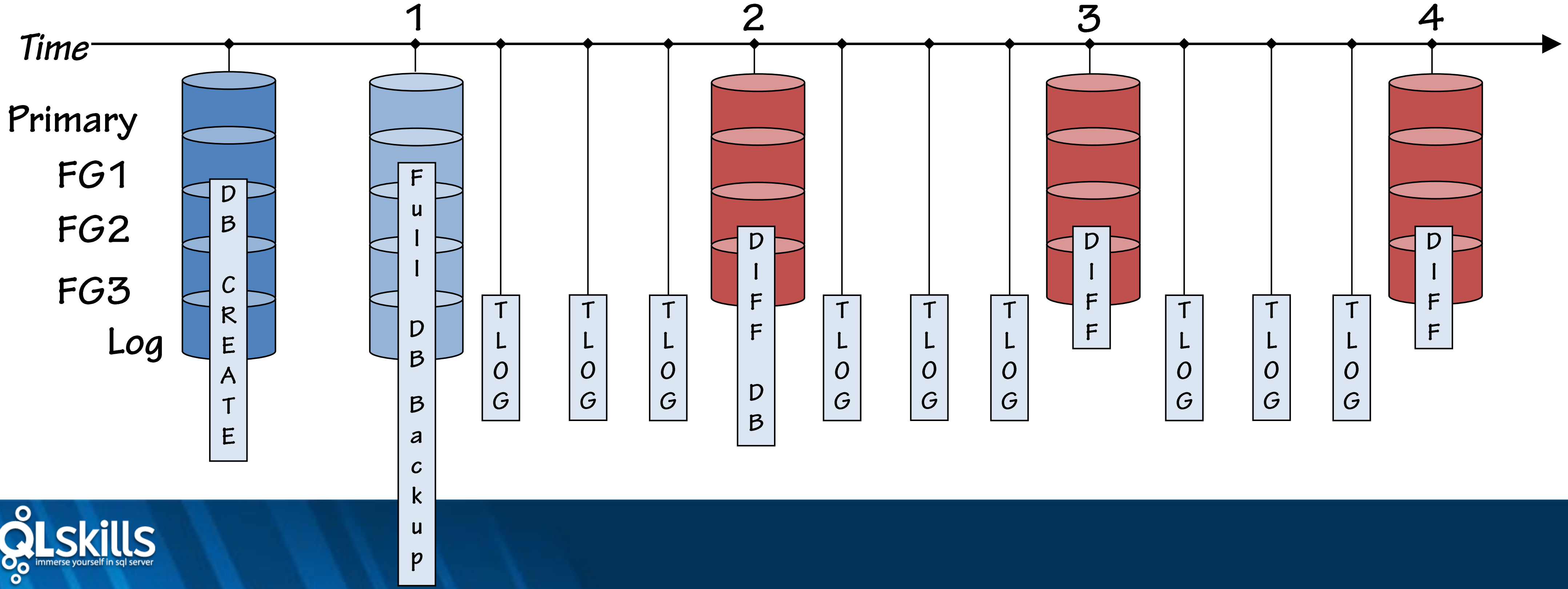
Full Database Backup plus Log Backups Strategy

- Up-to-point-of-crash recovery is possible with no data loss
- Can work around damaged full backup if prior log backups are still available
- Restoring a large number of log backups can take a significant amount of time so may need to include differential backups



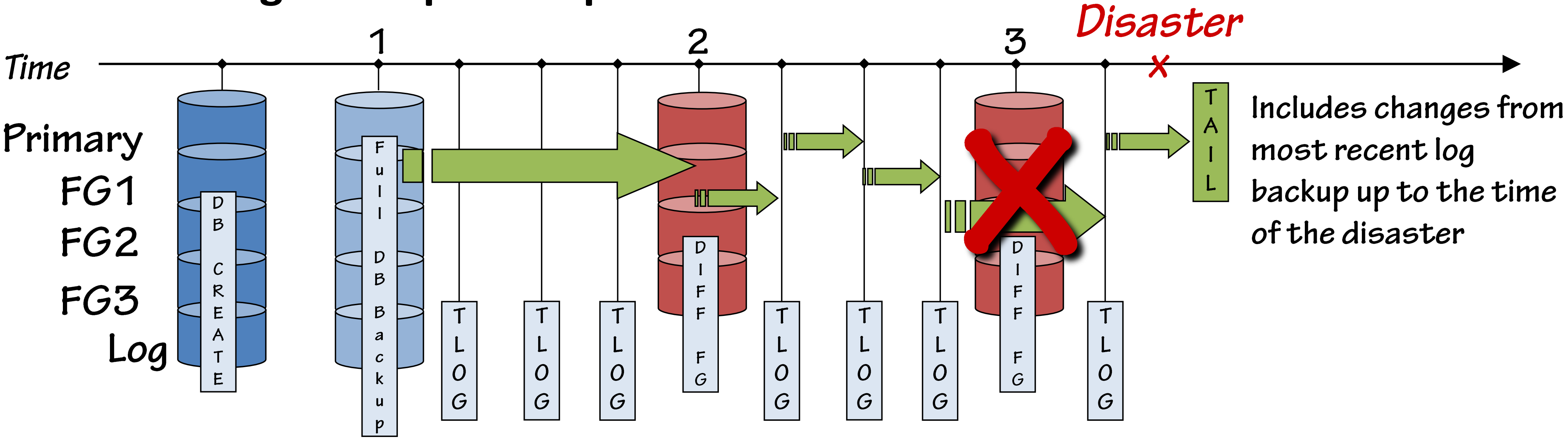
Full, Diff Database, and Log Backup Strategy

- Differential database backups are only the extents changed since the last full database backup
- Simplifies recovery time by allowing you to recover the database, the last differential, and only the logs after that, until the time of the disaster



Falling Back on Log Backups

- What if differential backup 3 was bad?
- Restore full (1), latest working differential (2) and all log backups up to the point of the failure (between 2 and 3) and finally the tail of the log
- So how long to keep backups?



How Long To Keep Backups?

- **Just because you took a full backup, doesn’t mean you should go delete the previous one**
    - What if the backup is corrupt, and you need to go back to the previous one… that you already deleted?
- **Same applies to differential and log backups**
- **But how long to keep them around?**
    - Depends on overall strategy, size of backups, storage available
    - We have financial clients in the US that must keep all backups around for several months, plus full backups for 7 years because of regulations
    - Your mileage will vary

Phases of Restore

- **When you restore a backup, what happens?**
- **Some or all of:**
  - File creation and initialization
  - Data and/or transaction log copy
  - Crash recovery
    - Redo of the log for each backup restored
    - Undo of the log after the last backup is restored
- **It depends on type of backup and restore options used**

Improving Restore Performance

- **Restore as little as possible**
  - Page, file, filegroup
    - But everything must be restored to same point in time
    - Smaller restores than database usually not possible without log backups
- **Use instant file initialization**
- **Use backup compression**
- **Use parallelism through multiple backup devices and/or data files**
- **Use a faster I/O subsystem**
- **Avoid long-running transactions that require a lot of undo**

Your Strategy

- Everyone’s strategy is different
- Make sure your backup strategy reflects what you want to be able to restore, within the data-loss and downtime requirements
- Consider using backup compression to save space and restore time
- Always use backup checksums and verify your backups
- Have a disaster recovery plan worked out, and possibly even automated scripts to drive the restore process
- At the very least, have some kind of strategy!
- Test your backup strategy regularly!

References

- **TechNet Magazine article on Understanding SQL Server Backups**
  - http://technet.microsoft.com/en-us/magazine/dd822915.aspx
    - Formatting seems to be all messed up unfortunately
- **TechNet Magazine article on Recovering from Disasters Using Backups**
  - http://technet.microsoft.com/en-us/magazine/ee677581.aspx
- **SQL Server Magazine article on Advanced Backup and Restore Options**
  - http://www.itprotoday.com/database-backup-and-recovery/advanced-backup-and-restore-options

References

- **Pluralsight course: *SQL Server: Understanding and Performing Backups***
  - □ https://www.pluralsight.com/courses/sqlserver-understanding-performing-backups
- **Blog posts:**
  - □ http://www.sqlskills.com/BLOGS/PAUL/category/BackupRestore.aspx

Thank you!

Questions? Paul@SQLskills.com

SQLskills

immerse yourself in sql server