COMPUTERWORLD SNIA

SNW

Driving Innovation Through the Information Infrastructure

SPRING 2011

Best Practices: Protecting, Replicating and Deduplicating Mission-Critical Databases

Ashish Ray
Senior Director, Product Management
Oracle

Agenda

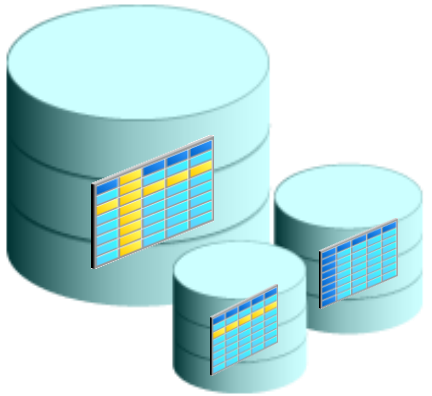
- Relational Database Architecture 101
- Best Practices: Data Protection
- Best Practices: Data Replication
- Considerations: Data Deduplication

Survey: Audience Background

- Show of hands – nature of profession:
  - Storage Administration
  - Database Administration
  - Systems Administration
  - Network Administration
  - IT Management
  - Vendor / Partner
  - Others

Relational Databases – Quick Intro

- Relational database: repository and management of data organized in a relational model *
- Database Management System (DBMS): the software infrastructure to maintain these key characteristics:
  - Structures: Well-defined objects to store data
  - Operations: Well-defined actions to access & update data & structure
  - Integrity rules: Well-defined rules to govern operations
- Data organized in a table: a two-dimensional relation with rows (tuples) and columns (attributes)
  - Each row in a table has the same set of columns
  - E.g. employee, department, salary tables

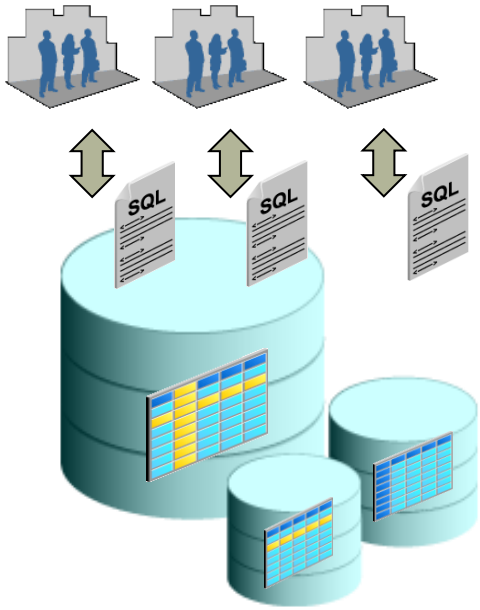


** Ref. "A relational model of data for large shared data banks", E. F. Codd, June 1970, http://portal.acm.org/citation.cfm?id=362685*

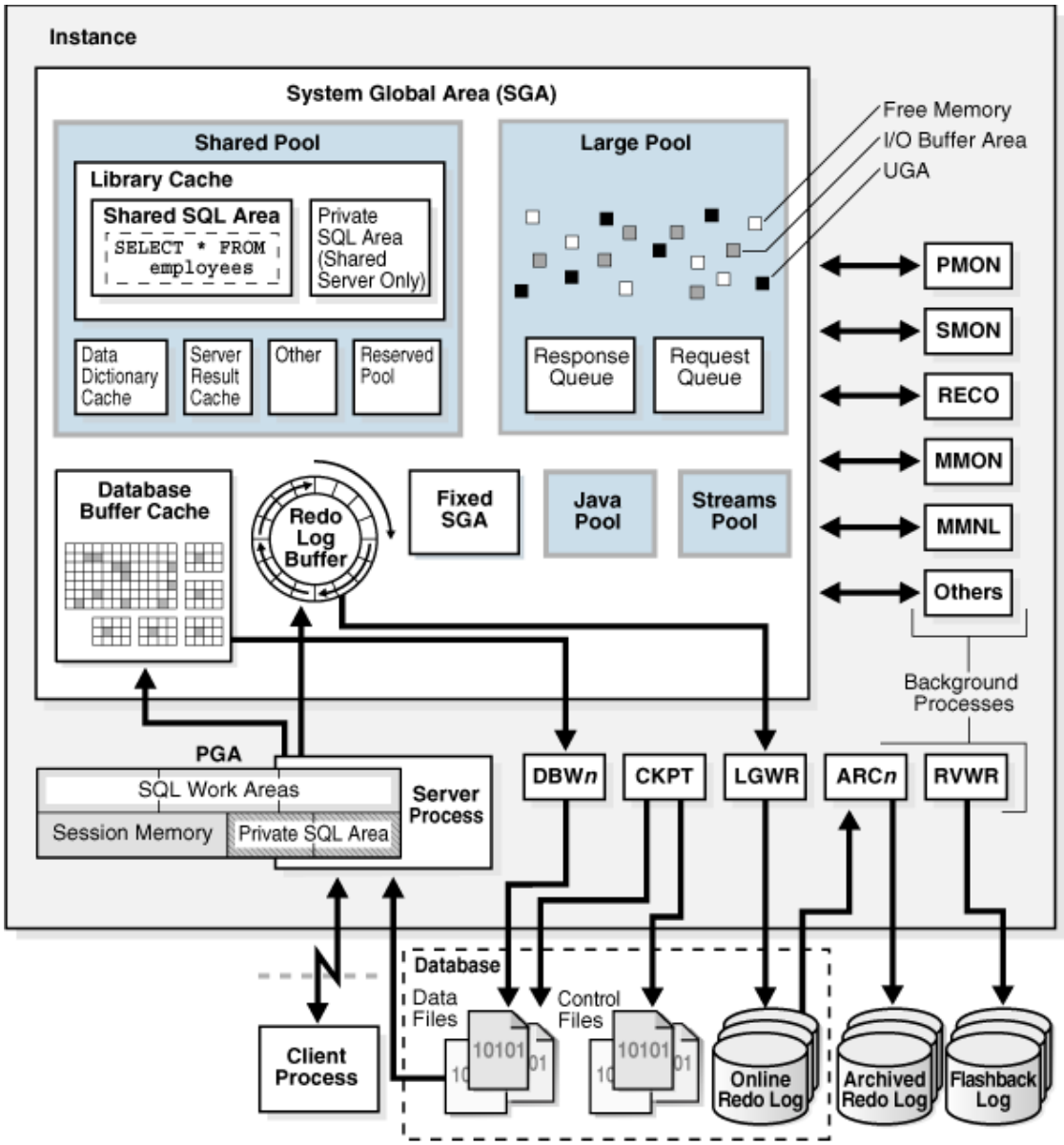
Transactions – The Secret Sauce

- Logical, atomic unit of work – composed of operations such that they all commit (i.e. applied to the database), or all roll back (i.e. undone)
  - Basic example: transfer $100 from savings to checking:
    - **Subtract** $100 from savings (**read** balance, **subtract** 100, **write** new balance)
    - **Add** $100 to Checking (**read** balance, **add** 100, **write** new balance)
    - **Write** a record of the transaction on a journal
  - All component operations must commit or all abort = a transaction
  - Failures and concurrency make achieving this nontrivial

- Transactions move database from one consistent state to another
  - Multiple users can work concurrently without corrupting one another's data
  - Difference with traditional file system: while updating several files, if the system fails halfway through, then the files will not be consistent

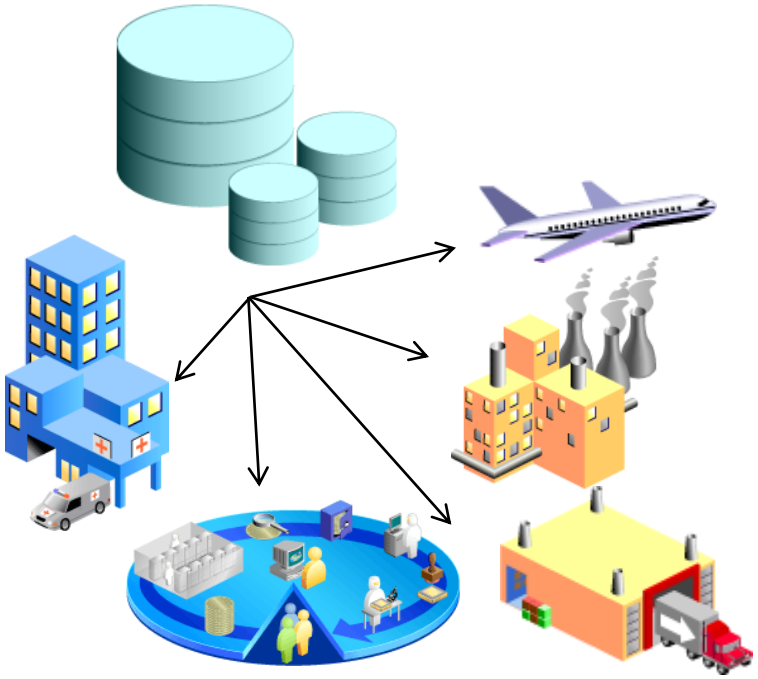


Database Architecture – Example



Ensuring Database Data Availability

- Modern databases support mission-critical businesses worldwide
- Data must always be available – it's a big deal!



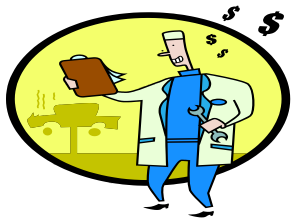
- For this presentation, Data Availability discussed across three dimensions
  - Data protection: focus on corruptions
  - Data replication: focus on RPO / RTO
  - Data deduplication: focus on optimization

Agenda

- Relational Database Architecture 101
- Best Practices: Data Protection
- Best Practices: Data Replication
- Considerations: Data Deduplication

Data Protection: Problem of Data Corruption

- Any component in the systems stack can fail and cause data corruptions*
  - Software: applications, middleware, database
  - Hardware: disk drives / controllers, HBAs, memory
  - Network: routers, switches, cables
  - Operational: human errors, bad installs & upgrades
- Increasing data volumes and complex I/O subsystems:
  - Impact of data corruptions is disastrous
  - Very hard to debug and diagnose



Ref. "Hard Disk Drives – the Good, the Bad & the Ugly", ACM Queue, Sep/Oct 2007, http://queue.acm.org/detail.cfm?id=1317403

Ref. "Silent Corruptions, CERN", http://fuji.web.cern.ch/fuji/talk/2007/kelemen-2007-C5-Silent_Corruptions.pdf

Data Corruption Example: Real-life Disaster

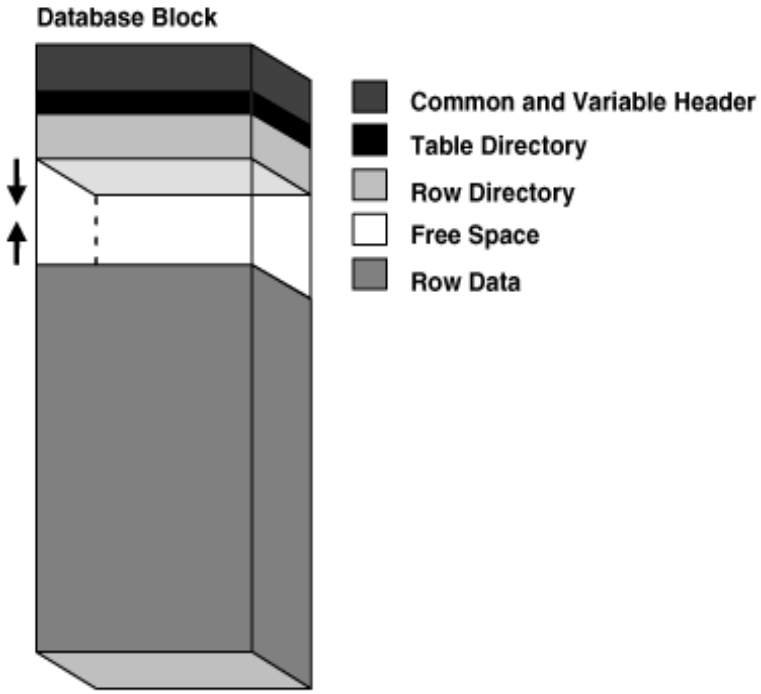
- Financial Services company: errors observed in the alert.log of the production database:
  - `Errors in file /opt/app/oracle/admin/dg/bdump/dg1.trc:`
  - `ORA-01186 : file 93 failed verification tests`
  - `ORA-01251 : Unknown File Header Version read for file number 93`

`ORA-01251 - Corrupted file header. This could be caused due to missed read or write or hardware problem or process external to oracle overwriting the information in file header.`

- Database crashed – applications down
  - Primary customer-facing applications for trade transaction confirmation, new accounts, and customer account information

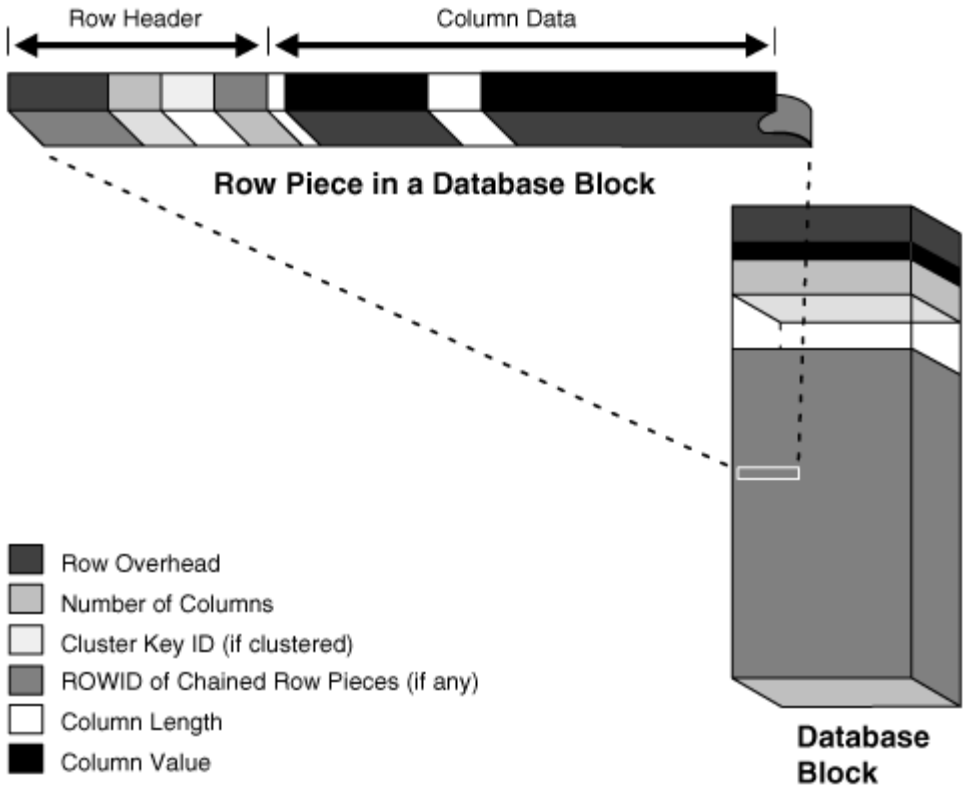
Database Block Format

- Database manages its logical storage in data blocks
  - Minimum unit of I/O
- A data block has a well-defined structure
  - Block header is kept consistent with payload, rows do not overlap, metadata in its place …
  - More than simple bits: can always verify logical consistency



Database Block Format … contd.

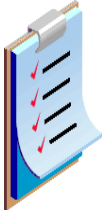
- Even the row (user) data is very carefully formatted within a database data block



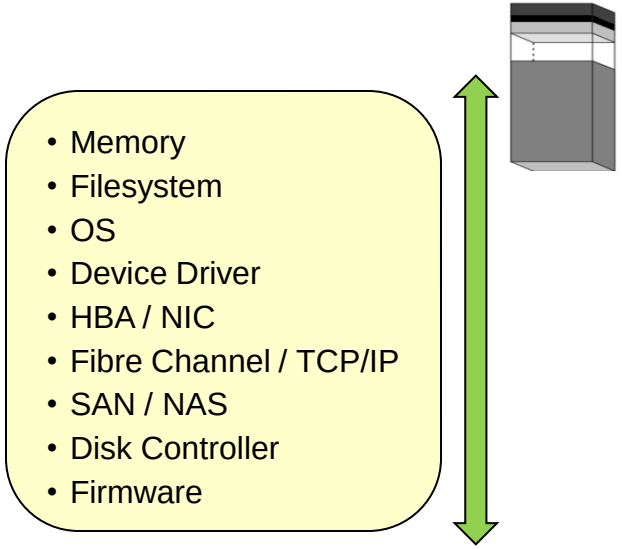
Requirements: Database Data Corruption Detection / Prevention

- Premise: provide constrained interfaces for better checks and better isolation
- Required: intrinsic checks, examples:
  - Data block checksum validation
  - Data block semantic check for logical consistency
  - Data block staleness check for lost writes
- Required: more than one copy of the data in secondary systems that are:
  - Loosely coupled
  - Fault isolated
  - Available for detection + repair

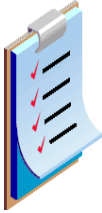
Best Practices Checklist



- Enable data corruption detection / correction techniques integrated with the database
  - Only the database system has intrinsic knowledge of its block structure
  - Only the database can perform true end-to-end validation as block traverses the system stack
  - Validation checks can be scaled across related processes such as transaction management, backup & recovery, mirroring, etc.



Best Practices Checklist … contd.



- Assess possible performance impact as you enable various database corruption checks, e.g.
  - Data Blocks / Index Blocks / both?
  - Checksum in-memory changes / before disk I/O?
  - System files / user files / both?
  - Full / partial logical consistency checks?
- Ensure underlying mirroring technology fully aware of database corruption algorithms
  - That way mirrored system also provides corruption detection, auto-repair, or fast failover if needed

  - ***Note: storage-mirroring technologies may propagate database block corruptions to target storage systems, rendering both the production database and the redundant copy unusable***

The Beauty Of Logs

- Fundamental structure of the database: transaction logs
  - Most crucial structure for data recovery
  - Contents: change data, undo data, schema / object management statements, etc.
  - Well-defined data fields, relationships

```
REDO RECORD - Thread:1 RBA: 0x00003e.0000d188.01b0 LEN: 0x07c8 VLD: 0x01
  SCN: 0x0000.00d6ca18 SUBSCN:  1 04/30/2007 21:06:42
  CHANGE #1 TYP:0 CLS:67 AFN:3 DBA:0x00c1fbb1 OBJ:4294967295 SCN:0x0000.00d6ca15 SEQ:  1 OP:5.2
  ktudh redo: slt: 0x0007 sqn: 0x00000000 flg: 0x000a siz: 120 fbi: 0
          uba: 0x00c05098.0004.01    pxid:  0x0000.000.00000000
```

- Why beautiful?
  - The well-defined structure and semantics of log data can be verified / validated independently of data files

```
ORA-00600: internal error code, arguments: [3020],..
ORA-10567: Redo is inconsistent with data block (file# 4, block# 315)
ORA-10564: tablespace TRANS1
ORA-01110: data file 4: 'O:\ORADATA\TBS01.DBF'
```

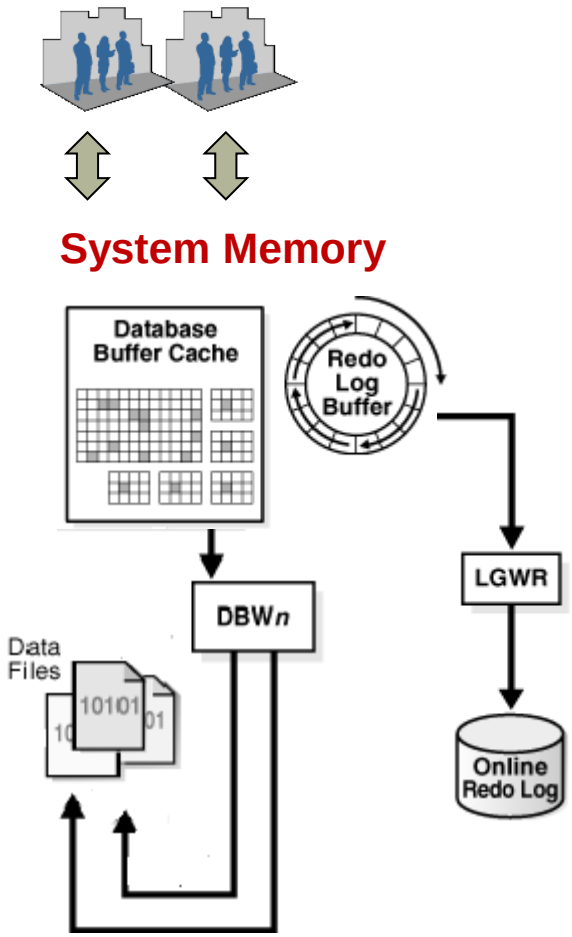
  - Forms the basis of sophisticated replication techniques that we will discuss next

Agenda

- Relational Database Architecture 101
- Best Practices: Data Protection
- Best Practices: Data Replication
- Considerations: Data Deduplication

Database Memory / Process Architecture

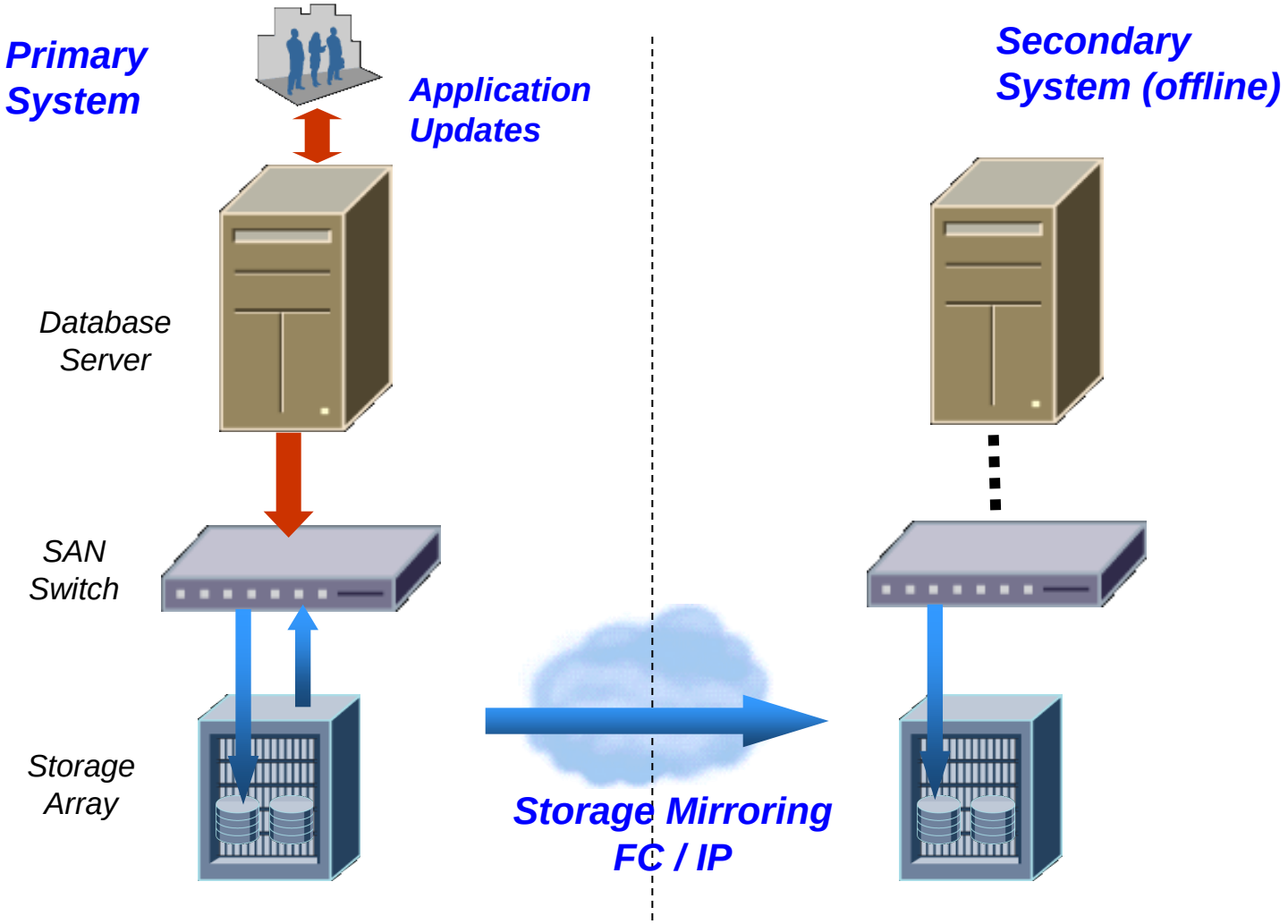
- Buffer Cache: memory area for copies of data blocks read from data files – reads / writes occur here
- Redo Log Buffer: circular in-memory buffer to reconstruct changes made to the database
- On transaction commit, the Log Writer process (LGWR) writes committed redo record to online redo logs (disk): atomic operation
- Database Writer process (DBWn) writes updated blocks in database buffer cache to data files in the background



Replication for Disaster Recovery

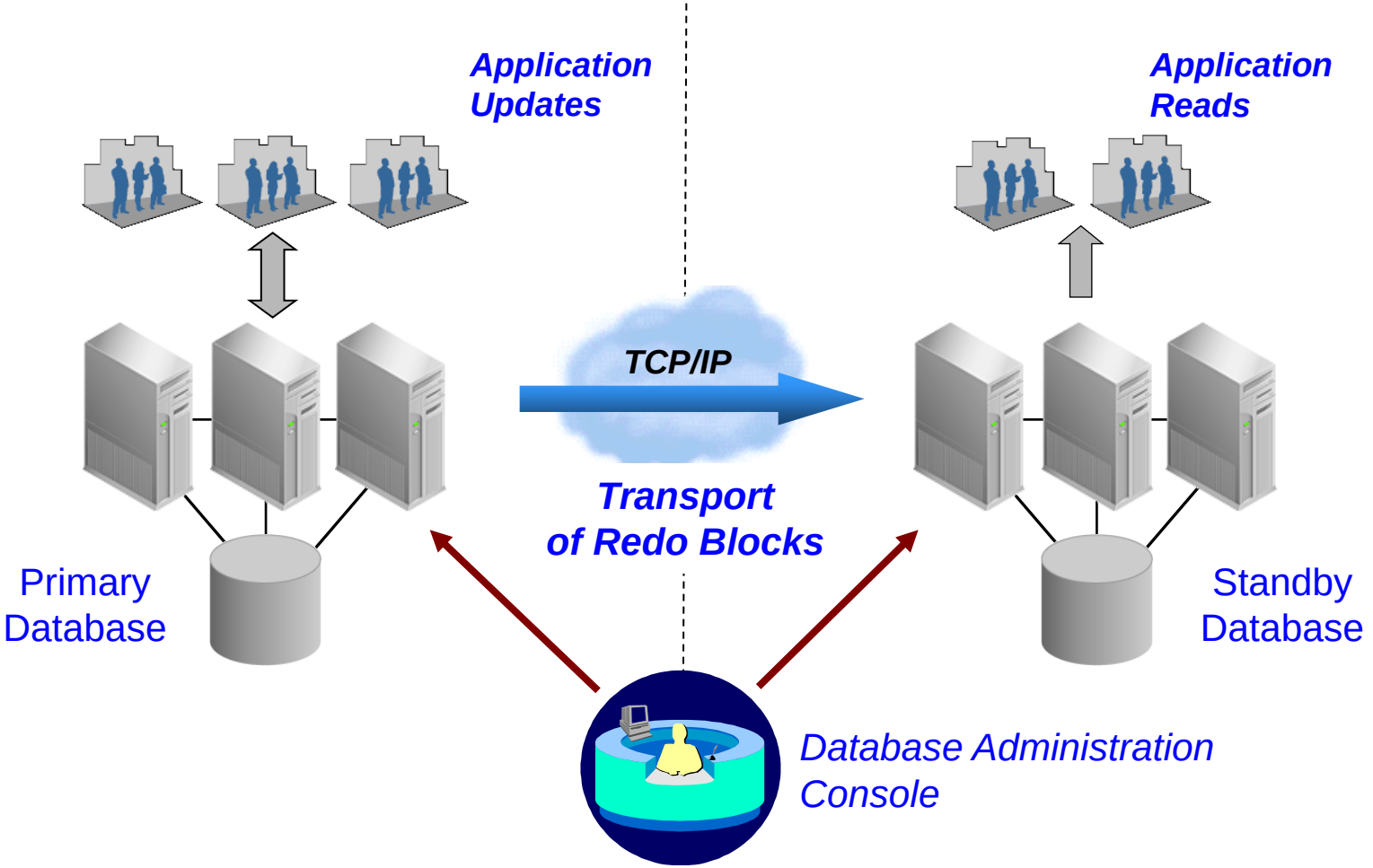
- The problem
  - Data corruptions / outages / disasters can wipe out production data
- The solution
  - Maintain synchronized replica of the data (locally / remotely)
- The challenge
  - Replica must be isolated from corruptions that impact the primary copy
- Two popular implementations
  - Storage-centric mirroring
  - Database-integrated replication

Storage Mirroring Configuration



Supports mix of database solutions, requires identical storage systems

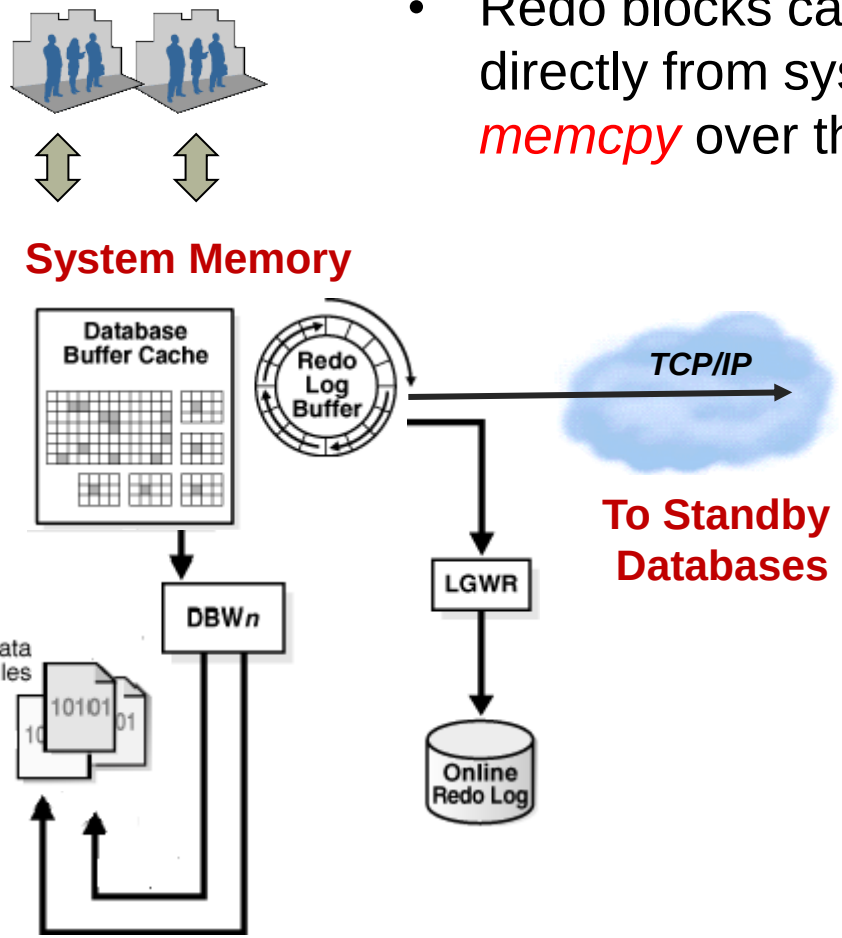
Database Integrated Replication



Supports mix of storage arrays, requires identical database systems

Database Integrated Replication

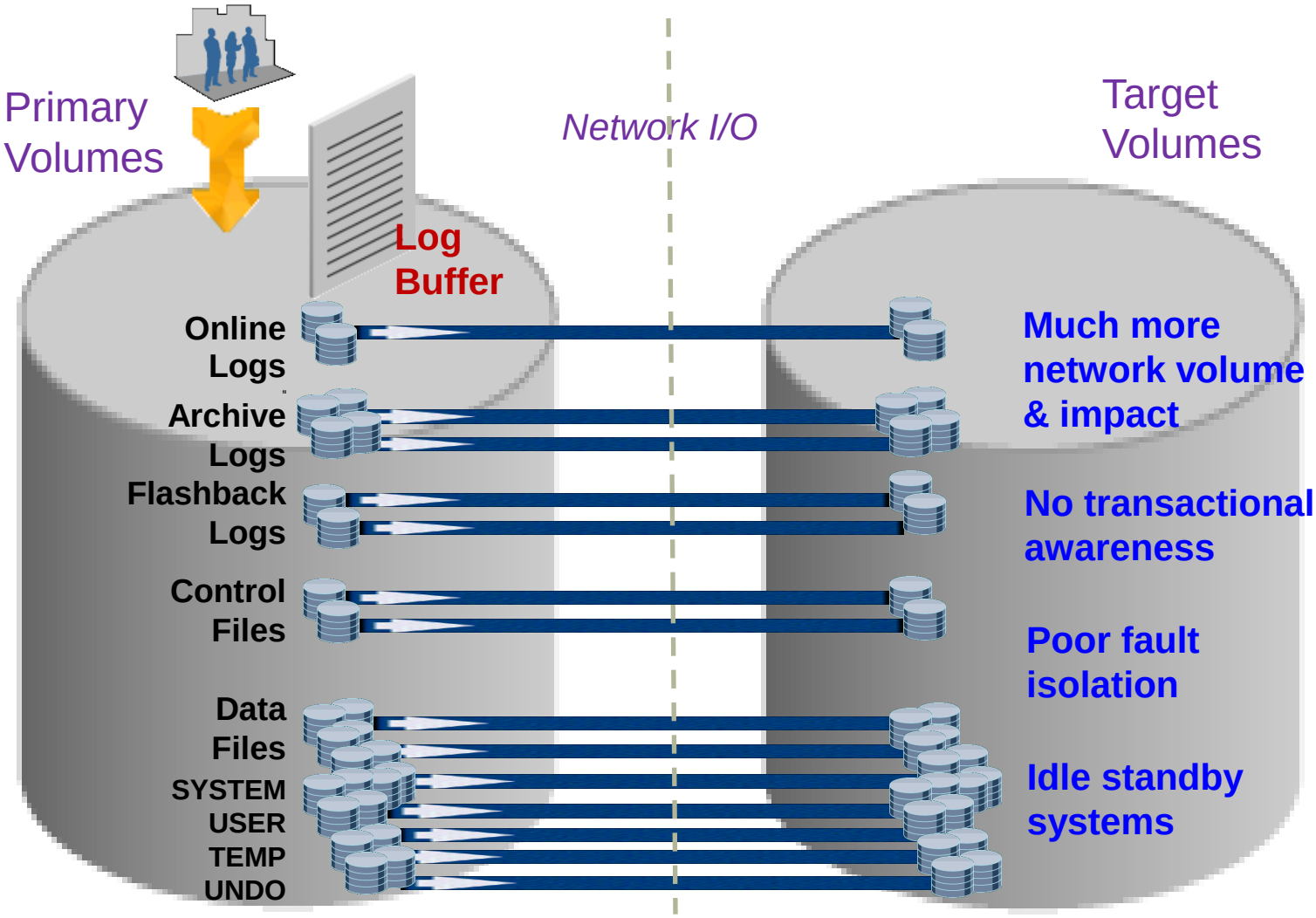
- Why is it a big deal?



- Redo blocks can be transmitted directly from systems memory: like a *memcpy* over the network

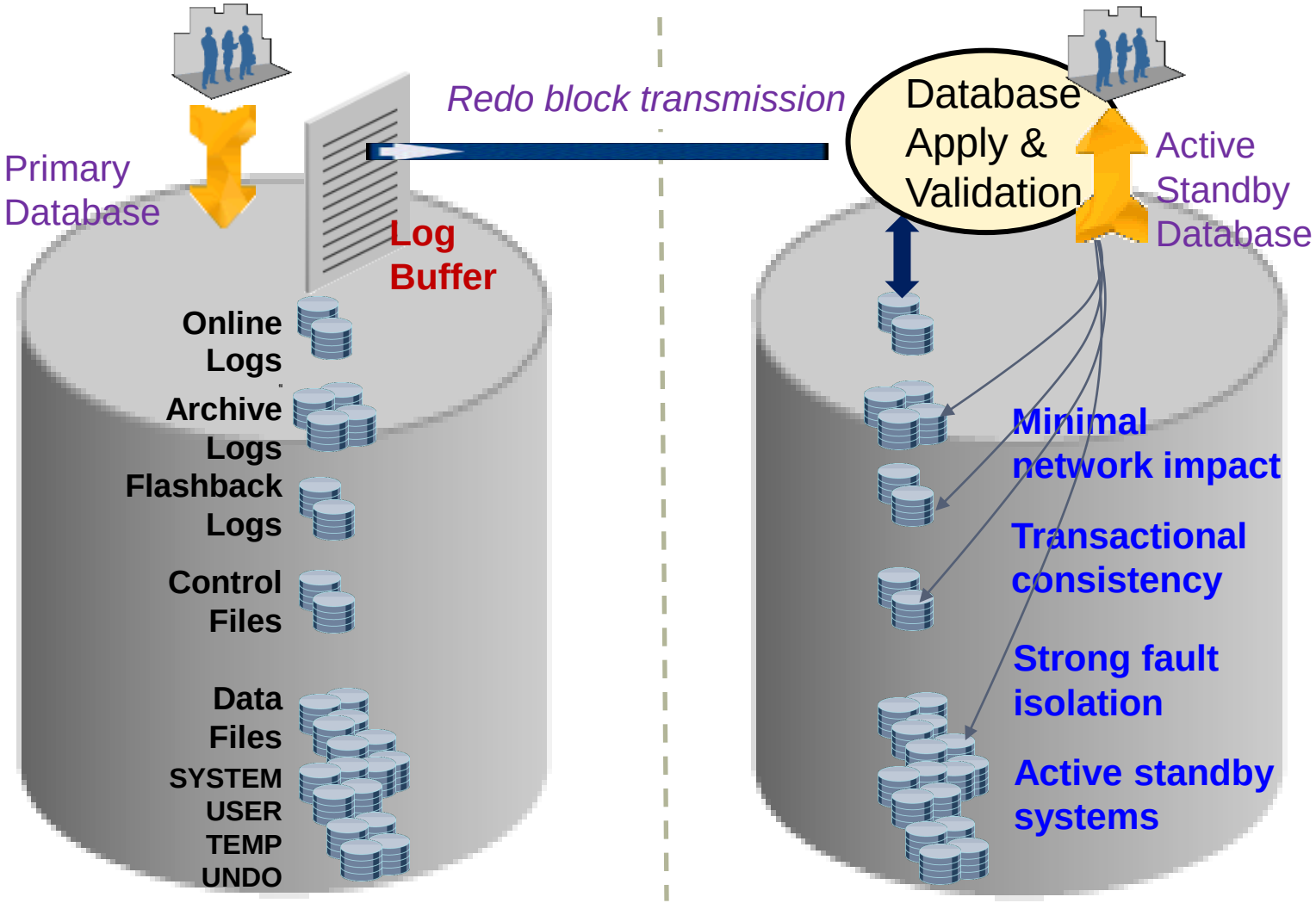
1. Better performance since no disk I/O
2. Better isolation from lower layer faults (e.g. storage)
3. Better network utilization: redo blocks reconstruct transaction changes, no other data sent over the network

Network Traffic with Storage Mirroring



Also: source database block corruptions propagated to target

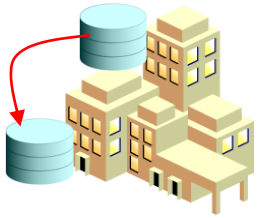
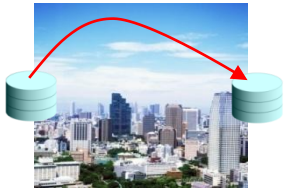
Network Traffic with DB Replication

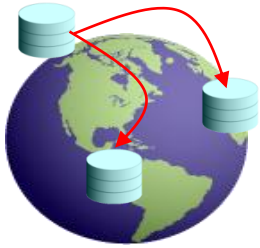


Also: standby database protected from primary block corruptions

Best Practices Checklist

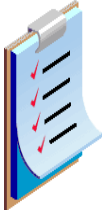
- For database environments, use database-integrated replication technologies



- For Recovery Point Objective (RPO):
  1. Use synchronous transport for zero data loss
     - Assess impact of network latency
     - If impact too high, use asynchronous transport
       - Data loss exposure should be minimal : e.g. not entire log files
       - Primary database should have minimal impact
     - Besides latency impact, there should be no other distance limitation
  2. For additional security / network optimization
     - Leverage built-in database encryption & compression

Best Practices Checklist … contd.



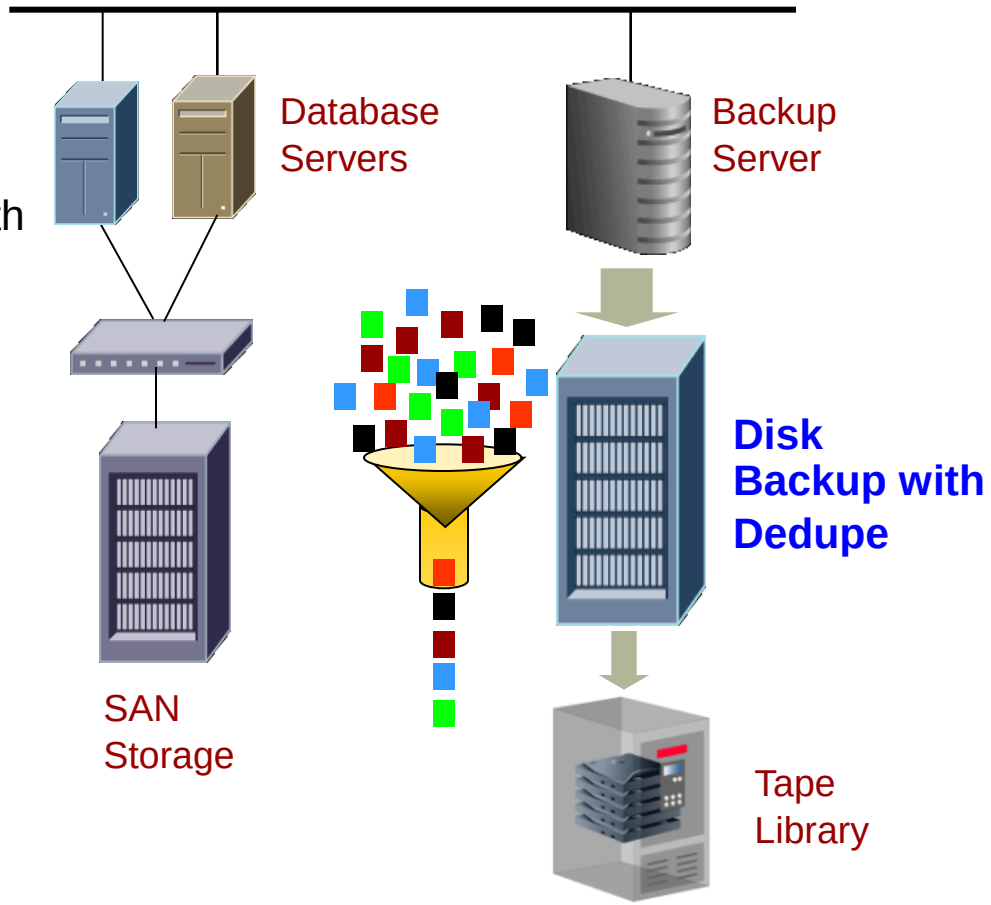
- For Recovery Time Objective (RTO):
  1. Failover process should be vey efficient
     - Very fast: 10-s of seconds
     - Support both manual & automatic failovers
     - Integrated with apps: not good if apps are stuck on old primary
  2. Standby database shouldn't be sitting idle – leverage it for queries / dev / testing / backups
     - Caution: Queries on standby should not show dirty data still uncommitted on primary database
     - Multiple standbys could be used to scale out read access – e.g. reader farms to handle hi-volume web access
- Implications:
  - Requires database logging to be enabled
  - Re-evaluate applications that run in no-logging mode given superior data protection

Agenda

- Relational Database Architecture 101
- Best Practices: Data Protection
- Best Practices: Data Replication
- Considerations: Data Deduplication

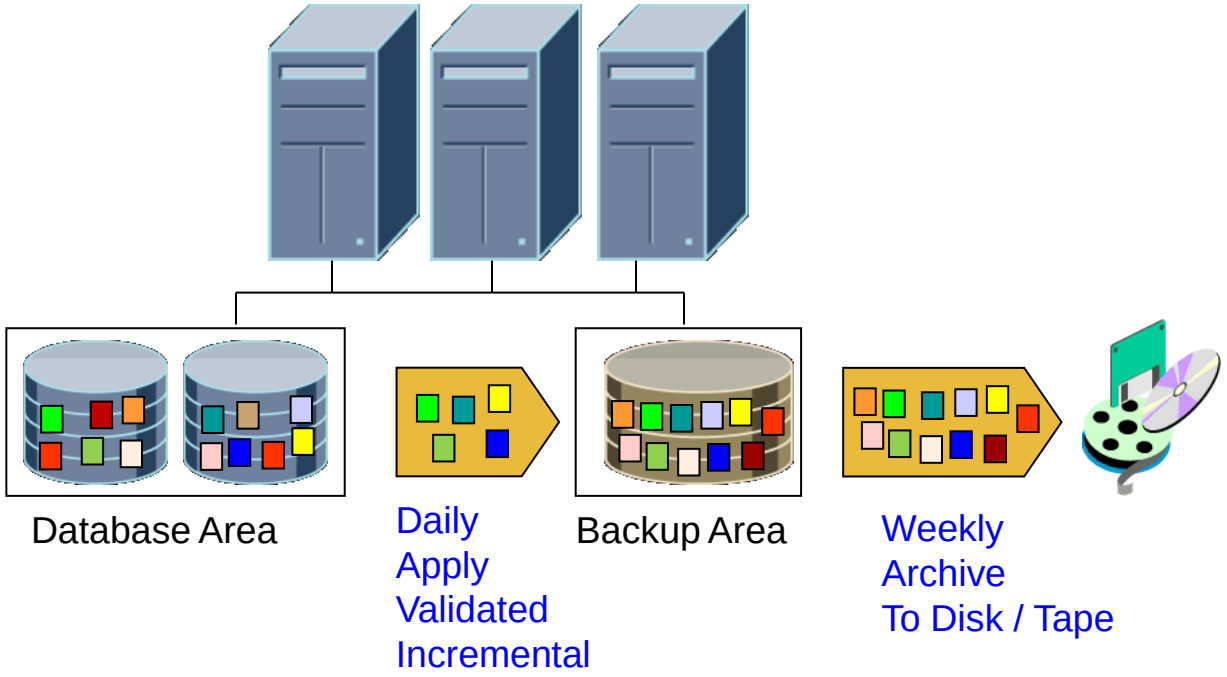
Data Deduplication 101

- Deduplication: Replace redundant backup data with pointers to shared copy
  - Lowers storage costs by reducing capacity requirements
  - Can be done at source, inline or post-process
  - Mileage varies for deduplication of database blocks



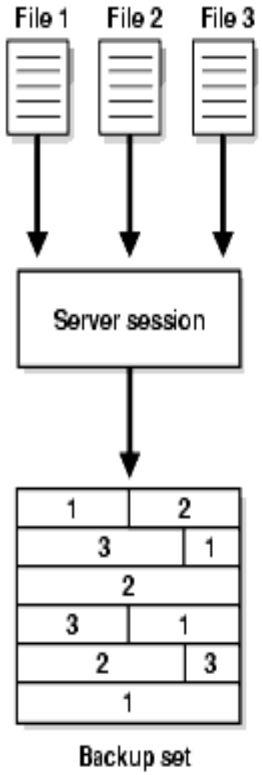
What About Database Backups?

- Deduplication ratios are minimal / unpredictable with widely used incremental backups:



Note: what's being backed up daily are only changed database blocks – no duplicate blocks in the daily backup stream

Database Backups & Deduplication: Other Considerations



- Multiplexing – number of data files simultaneously read by the database backup engine and then written into the same backup set
  - A higher multiplexing level could be optimal for database backup management but detrimental to effective deduplication

Database Backups & Deduplication: Other Considerations … contd.

- Compression
  - Several database integrated compression techniques:
    - OLTP data compression on user data
    - Columnar compression
    - Backup compression
  - Most will yield unpredictable dedupe performances on a data stream
- Encryption
  - Similar implications for dedupe with:
    - Column-level encryption
    - Physical datafile encryption
    - Backup encryption

Database Backups & Deduplication: So … No Hope?

- Well, it depends (*that's the answer no one likes to hear*)
- Better deduplication ratios possible:
  - With daily full backups
    - May not be practical, based on database size
  - By disabling database compression / encryption
    - Consistent with your security / capacity policies?
  - After customizing backup parameters
    - Need to balance database / dedupe best practices
- For storage optimization, do the testing + math:

Daily incremental backups + native compression	VS.	Daily full backups + 3rd party dedupe

Summary: Key Takeaways

1. Database data has sophisticated requirements around consistency, concurrency, atomicity, etc.
2. Solutions that work for unstructured data / file systems / email systems / documents etc. may not work in the same way in the DBMS world
3. Assess database-integrated solutions, and do the cost-benefit analysis