Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

Detailed set of interview questions and answers that align with SQL Server DBA responsibilities.

SQL Server DBA (2016 and above) – Frequently Asked Interview Questions & Detailed Answers

1. Installation & Configuration

Q: How do you perform a fresh installation of SQL Server 2019 on a new Windows Server, and what are some key configuration steps post-installation?

A:

1. **Preparation**
   - Verify hardware/software prerequisites (CPU, RAM, disk, OS version).
   - Install latest Windows updates.
   - Ensure proper service accounts are created with least privilege.
2. **Installation**
   - Launch SQL Server Setup, choose "New SQL Server stand-alone installation."
   - Select required features (Database Engine, SSIS, SSRS if needed).
   - Configure instance type (default/named).
   - Assign service accounts and set startup type.
   - Configure authentication mode (Windows or Mixed Mode).
   - Specify data directories (preferably separate drives for data, log, tempdb, backups).
3. **Post-Installation Configuration**
   - Apply latest CU/patch.
   - Configure **Max Memory** and **Max Degree of Parallelism (MAXDOP)**.
   - Set up **TempDB** with multiple data files.
   - Enable **Instant File Initialization**.
   - Configure default backup paths.
   - Create maintenance jobs (index rebuilds, stats updates, backups).
   - Implement monitoring alerts.

https://www.sqldbachamps.com

2. Performance Monitoring

Q: What tools and methods do you use to monitor SQL Server performance?

A:

- **Built-in Tools**:
  1. **SQL Server Management Studio (SSMS)** – Activity Monitor, Query Store.
  2. **Dynamic Management Views (DMVs)** – e.g., `sys.dm_exec_requests`, `sys.dm_os_wait_stats`.
  3. **Extended Events** – for tracking specific performance issues.
- **Performance Counters**:
  1. Buffer cache hit ratio.
  2. Page life expectancy (PLE).
  3. Disk latency (read/write).
  4. CPU utilization.
  5.

- **Third-Party Tools**: SolarWinds DPA, Redgate SQL Monitor, SentryOne.
- **Approach**:

1. Identify symptoms (slow queries, blocking, deadlocks).
2. Analyze execution plans.
3. Tune indexes, queries, and database configuration parameters.

3. Backup & Recovery

Q: Describe your backup strategy for a critical production database and how you'd perform a point-in-time recovery.
A:

- **Backup Strategy**:

1. Full backup **daily**.
2. Differential backup **every 4–6 hours**.
3. Transaction log backup **every 15 minutes**.
4. Store backups on separate storage + offsite/cloud copy.

- **Point-in-Time Recovery**:

1. Restore the most recent full backup **WITH NORECOVERY**.
2. Restore the latest differential backup **WITH NORECOVERY** (if used).
3. Restore transaction logs sequentially **WITH STOPAT** to the desired time.
4. Restore the last log **WITH RECOVERY**.

- **DR Considerations**: Test restores regularly; document RTO/RPO.

https://www.sqldbachamps.com

4. High Availability / Disaster Recovery

Q: Explain how Always On Availability Groups work and when you'd use them over Log Shipping.
A:

- **Always On AG**:

  - Requires Windows Server Failover Cluster (WSFC).
  - Allows multiple replicas (primary + up to 8 secondaries).
  - Supports automatic failover, read-only routing, backup from secondaries.

- **When to use**:

  - For low RTO and RPO.
  - When you need read-scale-out.
  - When database-level failover is sufficient (not instance-level like FCI).

- **Log Shipping**:

  - Simpler, lower-cost DR option.
  - No automatic failover.
  - Suitable for large gaps in RPO tolerance or cross-site DR without clustering.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

5. Security

Q: How do you secure SQL Server and ensure compliance?
A:

- **Access Control**:
  - Use Windows Authentication where possible.
  - Grant least privilege (principle of least privilege).
  - Create custom roles for application/service accounts.
- **Encryption**:
  - Enable Transparent Data Encryption (TDE).
  - Use Always Encrypted for sensitive columns.
  - Encrypt connections via SSL/TLS.
- **Auditing**:
  - Enable SQL Server Audit for critical events.
  - Track logins, failed login attempts, schema changes.
- **Compliance**:
  - Review permissions periodically.
  - Maintain audit logs securely for retention period.

6. T-SQL Optimization

https://www.sqldbachamps.com

Q: What steps do you take to optimize a slow-running stored procedure?
A:

1. Get the execution plan to identify bottlenecks (table scans, missing indexes).
2. Check for parameter sniffing issues and consider recompile hints or parameter optimization.
3. Optimize joins (use appropriate indexes, avoid unnecessary nested loops).
4. Reduce use of scalar functions inside SELECT statements.
5. Update statistics.
6. Apply index tuning (covering indexes, filtered indexes).
7. Consider splitting complex queries into smaller temp table operations.

7. Troubleshooting

Q: A query is blocking others in production. How do you identify and resolve it?
A:
Identify blocking session:

```
SELECT blocking_session_id, session_id, wait_type, wait_time, wait_resource
FROM sys.dm_exec_requests
WHERE blocking_session_id <> 0;
```

- Use `sp_who2` or `sys.dm_exec_sql_text` to get the exact query.
- Check if the blocking is due to locks, long-running transactions, or missing indexes.
- Resolve:
  - If safe, kill the blocking SPID (`KILL <spid>`).

- Tune query or adjust indexing.
- Consider isolation level changes (Read Committed Snapshot Isolation).

8. Automation

Q: How do you automate maintenance tasks in SQL Server?
A:

- **SQL Server Agent Jobs** for:
  - Full/diff/log backups.
  - Index rebuilds/reorgs.
  - Updating statistics.
- **PowerShell Scripts** for bulk administration (e.g., restoring multiple databases).
- **Maintenance Plans** for smaller environments.
- **Third-Party Tools** for advanced automation (e.g., Ola Hallengren's scripts).

9. Deployment & Upgrades

Q: How do you approach a SQL Server version upgrade with minimal downtime?
A:

1. Test upgrade in staging environment.
2. Run **Microsoft Data Migration Assistant** for compatibility issues.
3. Ensure backups are available and tested.
4. Use **rolling upgrade** in Always On AG environments.
5. Schedule downtime during low-traffic periods.
6. Verify application connectivity and post-upgrade performance.

https://www.sqldbachamps.com

10. Documentation

Q: Why is documentation important for DBAs, and what do you usually document?
A:

- Ensures quick recovery from incidents.
- Provides reference for troubleshooting and audits.
- **Typical Documentation**:
  - Server & instance configurations.
  - Backup & restore procedures.
  - HA/DR configurations.
  - Maintenance schedules.
  - User permissions & roles.

11. On-Call & Incident Response

Q: A production database went down at 2 AM. Walk me through your immediate actions.
A:

1. Check monitoring alerts and logs to identify the cause (error logs, Windows Event Viewer).
2. Verify if it's SQL Server service failure, OS-level issue, or storage/network failure.
3. Attempt to restart SQL Server if it's safe.
4. If DB corruption, run `DBCC CHECKDB` and restore from the last known good backup as a last option with client approvals.
5. Document the incident and communicate with stakeholders.

12. DevOps & CI/CD

Q: How do you integrate SQL Server database changes into a CI/CD pipeline?
A:

- Use source control for all database objects (e.g., Git).
- Implement automated builds with tools like Redgate SQL Change Automation or SSDT.
- Use pipeline tools (Azure DevOps, Jenkins) to run schema comparison, deploy changes to staging, run tests, and finally push to production after approval.
- Include rollback scripts for safety.

SQL Server DBA (2016+) – Quick Interview Cheat Sheet

1. Install & Configure

- Verify prerequisites, service accounts.
- Install features (DB Engine, SSIS, SSRS).
- Configure instance, authentication, directories.
- Post-install: Patch, set **Max Memory**, **MAXDOP**, multiple **TempDB** files, Instant File Init.

2. Performance Monitoring

https://www.sqldbachamps.com

- **Tools:** SSMS (Activity Monitor), DMVs, Query Store, Extended Events.
- **Metrics:** CPU, PLE, Buffer Cache Hit Ratio, disk latency.
- **3rd Party:** SolarWinds, Redgate, SentryOne.

3. Backup & Recovery

- Full daily, diff every 4–6 hrs, log every 15 min.
- Point-in-time: Restore full (NORECOVERY) → diff (NORECOVERY) → logs (STOPAT) → RECOVERY.
- Test restores; document RTO/RPO.

4. HA/DR

- **Always On AG:** WSFC, multiple replicas, auto failover, read-scale.
- **Log Shipping:** Simple, manual failover, low-cost DR.
- Use AG for low RTO/RPO, Log Shipping for cost-effective DR.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

5. Security

- **Auth:** Windows preferred, least privilege, custom roles.
- **Encrypt:** TDE, Always Encrypted, SSL/TLS.
- **Audit:** SQL Audit, failed logins, schema changes.
- Review permissions regularly.

6. T-SQL Optimization

- Review execution plan.
- Fix parameter sniffing.
- Optimize joins, add proper indexes.
- Update stats, avoid scalar UDFs.
- Use temp tables for complex queries.

7. Troubleshooting Blocking

- Find blocking SPID via DMVs/sp_who2.
- Check query, locks, indexes.
- Kill SPID if safe, tune query, adjust isolation level.

https://www.sqldbachamps.com

8. Automation

- SQL Agent jobs: backups, index maintenance, stats updates.
- PowerShell scripts for batch tasks.
- Ola Hallengren scripts for standardized maintenance.

9. Upgrades

- Test in staging, run Data Migration Assistant.
- Backup & verify.
- Use rolling upgrade in AG.
- Schedule downtime, post-upgrade validation.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

10. Documentation

- Server configs, backup/restore steps, HA/DR setups.
- Maintenance jobs, permissions, SOPs.
- Helps in audits, incident recovery.

11. On-Call Response

- Check alerts/logs.
- Identify root cause (service, OS, storage).
- Restart service if safe.
- For corruption, run `DBCC CHECKDB` & restore.
- Document & report.

12. DevOps / CI-CD

- DB objects in Git.
- Automated builds/deployments (SSDT, Redgate).
- Pipeline: schema compare → staging → tests → prod.
- Include rollback scripts.

https://www.sqldbachamps.com

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

Interview Questions & Answers – SQL Server Installation, Configuration, Upgrade, and Management (2016+)

1. Installation

Q: Walk me through the steps of installing SQL Server 2019 on a new Windows Server.
A:

1. **Preparation**
   - Verify **hardware & OS requirements** (CPU, RAM, disk, .NET version).
   - Create **dedicated domain service accounts** (DB Engine, Agent, SSIS).
   - Apply latest Windows updates.
   - Plan **storage layout** (Data, Log, TempDB, Backups on separate volumes).
2. **Installation Process**
   - Run **Setup.exe** → "New SQL Server stand-alone installation."
   - Select required features (Database Engine, Full-Text Search, SSIS, etc.).
   - Choose **default** or **named instance**.
   - Configure service accounts + startup types.
   - Choose authentication mode (Windows or Mixed) and add sysadmins.
   - Set default file locations (separate drives for performance).
3. **Post-Installation**
   - Install latest **Cumulative Updates (CUs)**.
   - Set **Max Memory** and **MaxDOP**.
   - Configure **TempDB** with multiple equal-sized data files.
   - Enable **Instant File Initialization** for faster growth.
   - Set up monitoring alerts (SQL Agent, email notifications).

https://www.sqldbachamps.com

2. Configuration Best Practices

Q: What key configurations do you apply after installing SQL Server?
A:

- **Memory:** Set `max server memory` to leave OS headroom.
- **Parallelism:** Configure `MAXDOP` based on CPU cores (generally ≤8).
- **TempDB:** Multiple data files, pre-sized, no auto-growth issues.
- **Database File Growth:** Fixed MB growth instead of percentage.
- **Backups:** Define default backup directory, enable compression.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

- **Error Logs:** Increase retention for historical troubleshooting.
- **Monitoring:** Enable alerts for severity ≥17, long-running queries, low disk space.

3. Instance Management

Q: How do you manage multiple SQL Server instances in a mixed prod/non-prod environment?
A:

- Use **SQL Server Central Management Server (CMS)** to manage all servers from one SSMS.
- Maintain **separate configurations** for prod and non-prod (different ports, naming standards).
- Apply changes in **non-prod first**, test, then deploy to prod.
- Use **Policy-Based Management** for enforcing standard configurations.
- Schedule maintenance windows for non-prod during working hours, prod during off-peak hours.

4. Upgrade Planning

Q: What steps do you take to upgrade from SQL Server 2016 to SQL Server 2019?
A:

1. **Assessment**
   - Use **Microsoft Data Migration Assistant (DMA)** to find compatibility issues.
   - Check application vendor certification for SQL 2019.
   - Validate hardware & OS compatibility.
2. **Testing**
   - Restore latest prod backup to a staging server with SQL 2019.
   - Test all application queries, SSIS packages, jobs, and linked servers.
3. **Execution**
   - Take full backup & log backup before upgrade.
   - If using Always On AG, perform **rolling upgrade**.
   - For standalone instances, perform in-place or side-by-side migration.
4. **Post-Upgrade**
   - Update database compatibility level to 150 (SQL 2019).
   - Update statistics, rebuild indexes.
   - Monitor performance for baseline deviations.

https://www.sqldbachamps.com

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

5. Upgrade Method Decision

Q: When would you choose a side-by-side upgrade over an in-place upgrade?
A:

- **Side-by-Side:**
  - Zero downtime is critical.
  - You want rollback capability.
  - Major hardware/OS changes.
  - Migration to a new data center or cloud.
- **In-Place:**
  - Minimal hardware changes.
  - No major schema/application modifications.
  - Short maintenance window available.

6. Production vs. Non-Production Considerations

Q: How do you manage configurations differently between prod and non-prod environments?
A:

- **Security:** Stricter access in prod, read-only access for most users; non-prod allows developers more privileges.
- **Performance Settings:** Prod tuned for maximum throughput; non-prod tuned for flexibility and testing.
- **Monitoring:** 24x7 alerts for prod, basic alerts for non-prod.
- **Data Masking:** Use masked or anonymized data in non-prod to meet compliance.
- **Maintenance Jobs:** Prod has more frequent index/stat updates; non-prod may run them less often.

https://www.sqldbachamps.com

7. Post-Upgrade Validation

Q: What steps do you take immediately after upgrading a SQL Server instance?
A:

- Verify service startup and connectivity.
- Check SQL Agent jobs, linked servers, and SSIS packages.
- Run `DBCC CHECKDB` on all databases.
- Update database compatibility level.
- Monitor CPU, memory, waits, and error logs.
- Run performance comparison against pre-upgrade baseline.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

8. Troubleshooting Installation/Upgrade Failures

Q: If a SQL Server upgrade fails halfway, how do you recover?

A:

- Review `Summary.txt` and detailed setup logs in `%ProgramFiles%\Microsoft SQL Server\...`.
- If in-place upgrade: Attempt repair using SQL Server setup.
- If database files are intact: Install new instance and attach/restore databases.
- If Always On AG: Failover to secondary and retry upgrade on primary later.
- Communicate issue to stakeholders and initiate rollback plan if needed.

https://www.sqldbachamps.com

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

SQL Server DBA Quick Recall – Install, Configure, Upgrade, Manage

Install (Fresh Setup)

- **Prep:** Check hardware/OS, create service accounts, plan storage layout.
- **Install:** Select features, configure authentication, set file paths.
- **Post-Install:** Apply CU, set **Max Memory**, **MAXDOP**, multiple TempDB files, Instant File Init, monitoring alerts.

Configuration Best Practices

- Max memory & MAXDOP tuning.
- Separate drives for Data, Log, TempDB, Backups.
- Pre-size files, fixed MB growth.
- Enable backup compression.
- Configure error log retention & alerts.

Managing Multiple Instances

- Use Central Management Server (CMS).
- Separate configs for prod/non-prod.
- Apply/test changes in non-prod first.
- Policy-Based Management for standards.

https://www.sqldbachamps.com

Upgrade Planning

- **Assessment:** Use DMA for compatibility, vendor certification.
- **Testing:** Restore prod backup to staging, run app tests.
- **Execution:** Backup before upgrade, rolling upgrade for AGs.
- **Post-Upgrade:** Update compatibility level, rebuild stats/indexes, monitor performance.

Side-by-Side vs In-Place

- **Side-by-Side:** New hardware, rollback needed, minimal downtime.
- **In-Place:** No hardware change, small maintenance window.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

Prod vs Non-Prod

- Stricter access in prod, masked data in non-prod.
- More aggressive monitoring in prod.
- Different maintenance schedules.

Post-Upgrade Validation

- Check services, connectivity, jobs, linked servers.
- Run DBCC CHECKDB.
- Monitor CPU/memory/waits.
- Compare to pre-upgrade baseline.

If Upgrade Fails

- Check setup logs (`Summary.txt`).
- Repair install or reattach databases to fresh instance.
- Failover to AG secondary if possible.
- Communicate & follow rollback plan.

https://www.sqldbachamps.com

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

Interview Questions & Answers – SQL Server Performance Monitoring & Tuning

1. Performance Monitoring Basics

Q: What are the main tools you use to monitor SQL Server performance?
A:

- **SSMS Tools**: Activity Monitor, Query Store, Extended Events.
- **DMVs**:
  - `sys.dm_exec_requests` → Active sessions.
  - `sys.dm_exec_query_stats` → Top resource-consuming queries.
  - `sys.dm_os_wait_stats` → Wait bottlenecks.
- **PerfMon Counters**:
  - CPU: `% Processor Time`.
  - Memory: Page Life Expectancy (PLE).
  - Disk: Avg. Disk sec/Read & Write.
- **Third-Party**: SolarWinds DPA, Redgate SQL Monitor, SentryOne.

2. Diagnosing a Slow-Running Query

https://www.sqldbachamps.com

Q: How do you troubleshoot a slow query in SQL Server?
A:

1. Check **execution plan** for scans, missing indexes, expensive operators.
2. Review **I/O stats** (`SET STATISTICS IO, TIME ON`).
3. Look for **parameter sniffing**; use `OPTION (RECOMPILE)` if necessary.
4. Update statistics (`UPDATE STATISTICS` or `sp_updatestats`).
5. Optimize joins, filter early, avoid `SELECT *`.
6. Add covering or filtered indexes where beneficial.

3. Wait Statistics

Q: How do you use wait statistics to diagnose performance issues?
A:
Query:

```
SELECT wait_type, wait_time_ms, signal_wait_time_ms
FROM sys.dm_os_wait_stats
ORDER BY wait_time_ms DESC;
```

-
- Example interpretations:

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

- PAGEIOLATCH_* → Disk I/O bottleneck.
- CXPACKET → Parallelism issues (check MAXDOP).
- LCK_* → Locking/blocking issues.

Clear wait stats after maintenance to monitor fresh workload:

```
DBCC SQLPERF('sys.dm_os_wait_stats', CLEAR);
```

4. Index Tuning

Q: How do you identify missing or unused indexes?
A:
Missing:

```
SELECT * FROM sys.dm_db_missing_index_details;
```

Unused:

```
SELECT * FROM sys.dm_db_index_usage_stats
WHERE user_seeks = 0 AND user_scans = 0 AND user_lookups = 0;
```

- Remove truly unused indexes to reduce write overhead.
- Add indexes carefully after testing impact.

https://www.sqldbachamps.com

5. Blocking & Deadlocks

Q: How do you identify and resolve blocking issues?
A:
Find blockers:

```
SELECT blocking_session_id, session_id
FROM sys.dm_exec_requests
WHERE blocking_session_id <> 0;
```

- Resolve:
  - Kill blocking SPID if safe.
  - Tune queries to reduce lock contention.
  - Use **Read Committed Snapshot Isolation** (RCSI) if appropriate.
- For deadlocks: Enable trace flag 1222 or use Extended Events to capture details.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

6. Query Store

Q: How can Query Store help in performance tuning?
A:

- Tracks query performance over time.
- Identifies regressions after upgrades/deployments.
- Allows **forcing a previous good plan** for problematic queries.
- Useful for long-term trend analysis, unlike real-time monitoring.

7. TempDB Performance

Q: What are best practices for TempDB to improve performance?
A:

- Multiple equal-sized data files (generally 1 per CPU core up to 8).
- Place TempDB on fast storage (SSD).
- Pre-size files to avoid auto-growth.
- Enable Instant File Initialization (for data files).

8. Statistics & Maintenance

https://www.sqldbachamps.com

Q: How do outdated statistics affect performance?
A:

- Leads to poor execution plans (incorrect row estimates).

Fix by updating regularly:

```
UPDATE STATISTICS TableName WITH FULLSCAN;
```

- Automate with index maintenance jobs (e.g., Ola Hallengren scripts).

9. Baseline Monitoring

Q: Why is a performance baseline important?
A:

- Helps differentiate between normal load and abnormal spikes.
- Use PerfMon + DMVs over time to log CPU, I/O, waits, and query response times.
- Baselines allow data-driven tuning decisions.

10. Applying Tuning Recommendations

Q: How do you prioritize tuning recommendations from a performance report?
A:

- Start with **highest-impact queries** (based on CPU, reads, writes).

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

- Fix low-cost, high-gain improvements first (e.g., missing index on a critical query).
- Validate changes in staging before production.
- Monitor after applying to ensure actual improvement.

SQL Server Performance Monitoring & Tuning – Quick Recall Cheat Sheet

1. Monitoring Tools

- **SSMS:** Activity Monitor, Query Store, Extended Events.
- **DMVs:**
  - `sys.dm_exec_requests` → Active queries.
  - `sys.dm_exec_query_stats` → Top resource consumers.
  - `sys.dm_os_wait_stats` → Wait bottlenecks.
- **PerfMon Counters:** CPU %, PLE, Disk latency.
- **3rd Party:** SolarWinds, Redgate, SentryOne.

2. Slow Query Diagnosis

https://www.sqldbachamps.com

- Check execution plan (scans, missing indexes).
- `SET STATISTICS IO, TIME ON`.
- Fix parameter sniffing.
- Update stats.
- Optimize joins, avoid `SELECT *`.
- Add covering/filtered indexes.

3. Wait Stats Basics

- `PAGEIOLATCH_*` → Disk I/O issue.
- `CXPACKET` → Parallelism issue (adjust MAXDOP).
- `LCK_*` → Locking/blocking.

Clear stats:

```
DBCC SQLPERF('sys.dm_os_wait_stats', CLEAR);
```

-

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

4. Index Tuning

- Missing: `sys.dm_db_missing_index_details`.
- Unused: `sys.dm_db_index_usage_stats`.
- Remove unused, add tested indexes.

5. Blocking/Deadlocks

- Find blockers: `sys.dm_exec_requests`.
- Resolve: Kill SPID (if safe), tune queries, enable RCSI.
- Capture deadlocks: Extended Events / TF 1222.

6. Query Store Use

- Tracks performance over time.
- Detect regressions.
- Force stable plans if needed.

7. TempDB Best Practices

https://www.sqldbachamps.com

- Multiple equal-sized files (1 per CPU core, up to 8).
- SSD storage.
- Pre-size to avoid growth.

8. Statistics

Update regularly:

```
UPDATE STATISTICS TableName WITH FULLSCAN;
```

- Automate with maintenance jobs.

9. Baseline Monitoring

- Track CPU, I/O, waits over time.
- Compare to spot anomalies.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

10. Tuning Priority

- Focus on high-impact queries first.
- Apply quick wins early.
- Validate in staging, monitor after deployment.

Interview Questions & Detailed Answers – Backup, Recovery, DR & HA in SQL Server

1. Backup Types

Q: What are the main types of backups in SQL Server, and when would you use each?
A:

- **Full Backup:** Entire database (all data + part of transaction log). Use daily or weekly as the base backup.
- **Differential Backup:** Changes since the last full backup. Use to reduce restore time.
- **Transaction Log Backup:** All transactions since last log backup. Needed for **point-in-time recovery** (Full/Bulk-Logged recovery model).
- **File/Filegroup Backup:** Specific file/filegroup; useful for large DBs with partitioned data.
- **Copy-Only Backup:** Doesn't affect backup chain. Good for ad-hoc backups without breaking the sequence.
- **Tail-Log Backup:** Captures remaining log before restoring, usually after failure.

https://www.sqldbachamps.com

2. Backup Strategy Design

Q: How would you design a backup strategy for a critical 24x7 OLTP system?
A:

- **Recovery Model:** Full.
- **Schedule:**
  - Full backup daily (night).
  - Differential backups every 4–6 hours.
  - Transaction log backups every 15 minutes.
- **Storage:**
  - Backups on separate physical disk from DB.
  - Copy to offsite/cloud storage (Azure Blob, AWS S3).
- **Testing:**
  - Regular restore tests in staging.
  - Verify backups with `RESTORE VERIFYONLY`.

- **RPO/RTO:** Ensure meets business SLAs.

3. Point-in-Time Recovery

Q: How do you restore a database to a specific point in time?
A:

1. Restore **Full Backup** with `NORECOVERY`.
2. Restore latest **Differential Backup** (if used) with `NORECOVERY`.
3. Restore **Transaction Log Backups** sequentially with `STOPAT 'YYYY-MM-DD HH:MM:SS'`.
4. Final log restore with `RECOVERY`.

Example:

```
RESTORE DATABASE Sales FROM DISK = 'Full.bak' WITH NORECOVERY;
RESTORE DATABASE Sales FROM DISK = 'Diff.bak' WITH NORECOVERY;
RESTORE LOG Sales FROM DISK = 'Log1.trn' WITH STOPAT = '2025-08-15 10:30:00', RECOVERY;
```

4. Backup Compression

Q: Why enable backup compression, and what are the trade-offs?
A:

- **Benefits:** Smaller backup size, faster writes to disk.
- **Trade-offs:** Higher CPU usage during backup.
- **Best Practice:** Enable if CPU capacity allows or run during low-load times.

https://www.sqldbachamps.com

5. Always On Availability Groups

Q: Explain how Always On Availability Groups work in SQL Server.
A:

- Requires **Windows Server Failover Cluster (WSFC)**.
- Supports **multiple replicas** (primary + up to 8 secondaries).
- **Failover Modes:** Automatic (synchronous), Manual (asynchronous).
- Benefits: Automatic failover, read-only routing, backups from secondaries.
- Limitations: Works at **database level**, not instance level.

6. Always On vs Failover Cluster Instance (FCI)

Q: What's the difference between Always On AG and FCI?
A:

- **Always On AG:** Database-level HA, each node has its own storage. Supports readable secondaries.
- **FCI:** Instance-level HA, all DBs in instance failover together, shared storage. No readable secondaries.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

7. Log Shipping

Q: How does Log Shipping work, and when is it best used?
A:

- Automates backup, copy, and restore of transaction logs from primary to secondary.
- **Pros:** Simple, works across sites, low cost.
- **Cons:** No automatic failover.
- Best for **DR** with relaxed RPO/RTO requirements.

8. Replication

Q: Briefly explain replication types in SQL Server.
A:

- **Snapshot Replication:** Full copy of data at intervals.
- **Transactional Replication:** Real-time or near real-time changes; best for reporting.
- **Merge Replication:** Bi-directional changes; suited for occasionally connected systems.

9. DR Testing

Q: How do you test DR readiness for SQL Server?
A:

- Simulate failover for Always On AG or FCI.
- Restore latest backups in DR site.
- Test application connectivity & performance.
- Document RTO/RPO compliance.

https://www.sqldbachamps.com

10. Backup & Restore Automation

Q: How do you automate backups and restores?
A:

- SQL Server Agent Jobs (Ola Hallengren's scripts).
- PowerShell scripts for advanced scheduling & offsite copies.
- Maintenance Plans for smaller environments.

11. Backup Verification

Q: How do you verify a backup is usable?
A:

- Run `RESTORE VERIFYONLY` for quick validation.
- Perform **test restores** in non-prod regularly.
- Check backup file size and timestamps.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

12. Scenario Question

Q: Your primary Always On node fails — what do you do?

A:

1. Check WSFC status and AG health in SSMS/`sys.dm_hadr_cluster_members`.
2. If synchronous commit + automatic failover: AG will failover automatically.
3. If manual failover: Initiate failover via SSMS or `ALTER AVAILABILITY GROUP ... FAILOVER`.
4. Verify secondary is now primary, clients connected.
5. Investigate cause of failure before failing back.

SQL Server Database Security – Interview Q&A

1. Security Models

Q: What are the two main security levels in SQL Server?

A:

- **Server-Level Security:** Controls access to the SQL Server instance. Uses **logins** (SQL Server Authentication or Windows Authentication). Server-level permissions are assigned through **fixed server roles** (e.g., `sysadmin`, `securityadmin`).
- **Database-Level Security:** Controls access to objects inside a database. Uses **database users** mapped to logins, **database roles**, and **permissions** (`SELECT`, `INSERT`, `EXECUTE`, etc.).

https://www.sqldbachamps.com

2. Authentication Modes

Q: What authentication modes does SQL Server support?

A:

- **Windows Authentication:** Uses AD credentials; recommended for integrated security.
- **Mixed Mode:** Allows both Windows Authentication and SQL Server Authentication (local SQL logins).

3. Login vs User

Q: What's the difference between a login and a user in SQL Server?

A:

- **Login:** Server-level security principal that allows a connection to the SQL Server instance.
- **User:** Database-level security principal mapped to a login, allowing access to a specific database.

4. Roles in SQL Server

Q: Explain the difference between fixed and custom roles.

A:

- **Fixed Roles:** Predefined roles with fixed permissions (e.g., `db_owner`, `db_datareader`, `db_datawriter`).
- **Custom Roles:** User-created roles with specific permissions assigned based on business requirements.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

5. Principle of Least Privilege

Q: How do you apply the principle of least privilege in SQL Server?
A:

- Grant **only the permissions necessary** for a user's job role.
- Avoid granting `sysadmin` unless absolutely required.
- Use **roles** to group permissions rather than assigning them directly to users.

6. Row-Level Security (RLS)

Q: What is Row-Level Security in SQL Server?
A:

- A feature that **restricts access to rows** in a table based on a filter predicate.
- Implemented using **Security Policies** with **Inline Table-Valued Functions (TVFs)**.

Example:

```
CREATE SECURITY POLICY SalesFilter
ADD FILTER PREDICATE dbo.fn_SalesFilter(UserID) ON Sales
WITH (STATE = ON);
```

7. Transparent Data Encryption (TDE)

Q: How does Transparent Data Encryption work in SQL Server?
A:

https://www.sqldbachamps.com

- Encrypts data **at rest** (data files, log files, backups).
- Uses **Database Encryption Key (DEK)** protected by a server certificate.
- Protects against physical theft of DB files.

8. SQL Server Auditing

Q: How do you implement auditing in SQL Server?
A:

- Use **SQL Server Audit** feature to track login activity, schema changes, failed logins, etc.
- Create a **Server Audit** and **Server/Database Audit Specifications**.
- Store logs in a secure location for compliance.

9. Login Auditing

Q: How do you track failed login attempts?
A:

- Enable **Login Auditing** in server properties (failed, successful, or both).
- View results in the **SQL Server Error Log** or Windows Event Viewer.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

10. Schema & Object Permissions

Q: How do you grant permissions at schema level?
A:

```
GRANT SELECT, INSERT, UPDATE ON SCHEMA::Sales TO SalesRole;
```

This applies permissions to all objects in that schema.

11. Securing sa Account

Q: What's your approach to securing the `sa` account?
A:

- Rename it.
- Disable it if not used.
- Use strong password & avoid direct use for daily operations.

12. Contained Databases

Q: What is a contained database and how does it affect security?
A:

- A database that has all its metadata and users stored within itself (no server-level login mapping).
- Useful for migration and isolation but requires careful security planning.

https://www.sqldbachamps.com

13. Data Masking

Q: How do you mask sensitive data in SQL Server?
A:

- Use **Dynamic Data Masking (DDM)** to obscure data for non-privileged users.

Example:

```
ALTER TABLE Customers
ALTER COLUMN Email ADD MASKED WITH (FUNCTION = 'email()');
```

14. Security Best Practices

Q: What are SQL Server security best practices?
A:

- Use Windows Authentication wherever possible.
- Apply least privilege principle.
- Encrypt sensitive data (TDE, Always Encrypted).
- Monitor with SQL Audit and login auditing.
- Regularly review and revoke unused permissions.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

T-SQL, Stored Procedures, Triggers & Index Optimization – Interview Q&A

1. T-SQL Basics

Q: What's the difference between T-SQL and SQL?
A:

- **SQL** is the ANSI standard for querying and manipulating data.
- **T-SQL (Transact-SQL)** is Microsoft's extension to SQL that includes procedural programming features (variables, loops, error handling, system functions).

2. Query Optimization

Q: How do you optimize a slow T-SQL query?
A:

1. Review the **execution plan** for scans vs seeks, missing indexes, and key lookups.
2. Avoid `SELECT *`, retrieve only required columns.
3. Use proper indexes (clustered, non-clustered, covering).
4. Update statistics for accurate query plans.
5. Eliminate non-SARGable predicates (e.g., avoid functions on columns in `WHERE`).
6. Break complex queries into smaller steps if needed.

https://www.sqldbachamps.com

3. Stored Procedures Advantages

Q: Why use stored procedures instead of ad-hoc queries?
A:

- Precompiled execution plans for better performance.
- Enhanced security (parameterized queries reduce SQL injection).
- Easier maintenance and reusability.
- Reduced network traffic.

4. Stored Procedure Optimization

Q: How do you optimize a stored procedure?
A:

- Use **parameters** and avoid dynamic SQL unless necessary.
- Handle parameter sniffing (local variables or `OPTIMIZE FOR`).
- Use **SET NOCOUNT ON** to avoid extra messages.
- Minimize temp table and table variable usage.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

5. Parameter Sniffing

Q: What is parameter sniffing and how do you handle it?
A:

- SQL Server compiles a plan based on the first parameter values passed.
- Can cause poor performance for other parameter values.
- Solutions:
  - Use `OPTION (RECOMPILE)`.
  - Use local variables to break sniffing.
  - Create separate procedures for different value ranges.

6. Triggers

Q: When should you use triggers?
A:

- For enforcing business rules that **cannot** be handled via constraints.
- For auditing changes (INSERT, UPDATE, DELETE).
- For cascading changes to related tables.

Avoid: Complex logic or heavy processing inside triggers (affects DML performance).

https://www.sqldbachamps.com

7. Trigger Types

Q: What are the types of triggers in SQL Server?
A:

- **AFTER triggers:** Run after DML statement completes.
- **INSTEAD OF triggers:** Replace execution of the DML statement.
- **DDL triggers:** Fire on schema changes like CREATE, ALTER, DROP.

8. Index Basics

Q: What is the difference between clustered and non-clustered indexes?
A:

- **Clustered Index:** Sorts and stores the data rows in order of the index key; one per table.
- **Non-Clustered Index:** Separate structure with a pointer (RID or clustering key) to the data.

9. Covering Index

Q: What is a covering index?
A:

- An index that contains all the columns a query needs, avoiding lookups to the base table.
- Achieved by **including** non-key columns:

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator


```
CREATE NONCLUSTERED INDEX IX_Covering
ON Sales(OrderDate)
INCLUDE (CustomerID, TotalAmount);
```

10. Index Maintenance

Q: How do you maintain indexes for performance?
A:

- Rebuild if fragmentation > 30%.
- Reorganize if fragmentation is 5–30%.
- Update statistics after rebuild.

11. Common Performance Killers in T-SQL

- Using functions in WHERE clauses (`WHERE YEAR(OrderDate) = 2023`).
- Implicit conversions.
- Cursors instead of set-based operations.
- Excessive nested views.

12. Execution Plan Analysis

https://www.sqldbachamps.com

Q: How do you analyze execution plans?
A:

- Look for **Table Scans** (bad for large tables).
- Check for **Key Lookups** (can be fixed with INCLUDE).
- Identify **high-cost operators** (e.g., Sort, Hash Match).
- Check for missing index recommendations.

13. Dynamic SQL

Q: When should you use dynamic SQL in T-SQL?
A:

- When table or column names need to be dynamic.
- Must be parameterized to avoid SQL injection.

```
DECLARE @sql NVARCHAR(MAX) = 'SELECT * FROM ' + QUOTENAME(@TableName);
EXEC sp_executesql @sql;
```

14. Best Practices Summary

- Always write **SARGable** queries.
- Use proper indexes and keep them maintained.
- Avoid scalar functions in SELECT and WHERE.
- Use stored procedures for business logic.
- Limit use of triggers to essential logic.

SQL Server Troubleshooting – Interview Q&A

1. General Troubleshooting Approach

Q: How do you approach troubleshooting a SQL Server issue?
A:

1. **Identify the symptom** – e.g., slow queries, connection failures, blocking.
2. **Check logs** – SQL Server Error Log, Windows Event Viewer.
3. **Narrow down the cause** – Is it query-related, configuration-related, hardware-related?
4. **Reproduce the issue** if possible in a non-prod environment.
5. **Fix** using minimal-impact changes.
6. **Document** the cause and solution.

https://www.sqldbachamps.com

2. Tools for Troubleshooting

Q: What tools do you use for SQL Server troubleshooting?
A:

- **SQL Server Management Studio (SSMS)** Activity Monitor.
- **Dynamic Management Views (DMVs)** like `sys.dm_exec_requests`, `sys.dm_os_wait_stats`.
- **Extended Events / SQL Profiler** for capturing query activity.
- **Performance Monitor (PerfMon)** for CPU, memory, disk I/O.
- **sp_whoisactive** (community tool) for real-time session analysis.

3. Blocking vs Deadlock

Q: What's the difference between blocking and deadlocks?
A:

- **Blocking:** One session holds a lock and prevents another from proceeding until it's released.
- **Deadlock:** Two or more sessions block each other in a cycle, requiring SQL Server to terminate one.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

4. Identifying Blocking Sessions

Q: How do you identify blocking in SQL Server?
A:

- Use Activity Monitor in SSMS.
- Query DMVs:

```
SELECT
    blocking_session_id, session_id, wait_type, wait_time, wait_resource
FROM sys.dm_exec_requests
WHERE blocking_session_id <> 0;
```

5. Deadlock Troubleshooting

Q: How do you capture and resolve deadlocks?
A:

- Enable **trace flag 1222** to log deadlocks to the error log.
- Use **Extended Events** (`xml_deadlock_report`).
- Resolve by:
  - Reducing transaction scope.
  - Accessing objects in the same order across transactions.
  - Adding appropriate indexes.

https://www.sqldbachamps.com

6. Performance Bottlenecks

Q: How do you identify performance bottlenecks?
A:

- Check **wait statistics** (`sys.dm_os_wait_stats`).
- Monitor CPU, memory, and I/O in PerfMon.
- Analyze query plans for expensive operators.

7. High CPU Usage

Q: How do you troubleshoot high CPU usage in SQL Server?
A:

- Identify top CPU-consuming queries:

```
SELECT TOP 5
    total_worker_time/1000 AS CPU_ms,
    execution_count,
    query_hash,
    SUBSTRING(text, 1, 1000)
FROM sys.dm_exec_query_stats
CROSS APPLY sys.dm_exec_sql_text(sql_handle)
ORDER BY total_worker_time DESC;
```

- Tune the queries, update statistics, add indexes if needed.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

8. TempDB Issues

Q: What are common TempDB problems and fixes?
A:

- **Contention on allocation pages** – fix by creating multiple data files (1 per logical CPU up to 8).
- **Running out of space** – increase file size or move to faster storage.

9. Connection Issues

Q: How do you troubleshoot when an application can't connect to SQL Server?
A:

- Check if SQL Server service is running.
- Verify TCP/IP protocol is enabled.
- Ensure correct port (default 1433) is open in firewall.
- Test connectivity using `sqlcmd` or SSMS.

10. Query Timeout Issues

Q: What causes query timeouts and how do you fix them?
A:

- **Causes:** Poor query performance, blocking, missing indexes, high network latency.
- **Fix:** Tune query, reduce locks, adjust application timeout settings if necessary.

https://www.sqldbachamps.com

11. Coordinating with Development Teams

Q: How do you work with developers to resolve database issues?
A:

- Review execution plans and query logic together.
- Suggest indexing or query rewrites.
- Test changes in lower environments before production deployment.

12. Coordinating with Infrastructure Teams

Q: How do you work with infrastructure teams during DB issues?
A:

- Share performance metrics (CPU, disk I/O, memory usage).
- Request hardware diagnostics if needed.
- Ensure storage latency and network issues are ruled out.

13. Common Proactive Measures

- Set up monitoring alerts (blocking, deadlocks, low disk space).
- Keep indexes/statistics updated.
- Run regular health checks.

SQL Server Automation – Interview Q&A

1. SQL Server Agent Basics

Q: What is SQL Server Agent and why is it used?
A:

- SQL Server Agent is a Windows service that automates administrative tasks like backups, index maintenance, ETL processes, and monitoring.
- It uses **Jobs** (tasks) and **Schedules** to run them automatically without manual intervention.

2. Creating a SQL Server Agent Job

Q: How do you create a SQL Server Agent job?
A:

1. In SSMS, expand **SQL Server Agent → Jobs → New Job**.
2. Provide **Name** and **Description**.
3. Add **Steps** (T-SQL scripts, SSIS packages, PowerShell commands, etc.).
4. Set a **Schedule** (daily, weekly, monthly, custom).
5. Configure **Alerts** and **Notifications**.

https://www.sqldbachamps.com

3. Automating Backups

Q: How do you automate backups using SQL Server Agent?
A:

- Create a job with a T-SQL step:

```
BACKUP DATABASE MyDB
TO DISK = 'D:\Backups\MyDB_Full.bak'
WITH INIT, STATS = 10;
```

- Schedule it to run daily or as needed.

4. Maintenance Plans

Q: What are Maintenance Plans and when do you use them?
A:

- A **GUI-based tool** in SSMS for automating tasks like backups, index rebuilds, integrity checks.
- Ideal for smaller environments or when scripting isn't necessary.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

5. PowerShell for Automation

Q: How can PowerShell be used for SQL Server automation?
A:

- PowerShell can run scripts to:
  - Perform backups/restores.
  - Run DBCC checks.
  - Import/export data.
  - Manage security and permissions.

Example:

```
Invoke-Sqlcmd -ServerInstance "SQLServer01" -Database "MyDB" `
-Query "EXEC dbo.usp_IndexMaintenance;"
```

6. Combining SQL Agent with PowerShell

Q: Can SQL Server Agent execute PowerShell scripts?
A:

- Yes, by adding a job step with type **PowerShell** and specifying the script path or inline commands.

7. Automating Index & Statistics Maintenance

https://www.sqldbachamps.com

Q: How do you automate index maintenance?
A:

- Create a job to run Ola Hallengren's free maintenance scripts or your own stored procedure:

```
EXEC dbo.IndexOptimize
@Databases = 'USER_DATABASES',
@FragmentationLow = NULL,
@FragmentationMedium = 'INDEX_REORGANIZE',
@FragmentationHigh = 'INDEX_REBUILD';
```

8. Error Handling in Automated Jobs

Q: How do you handle failures in automated jobs?
A:

- Configure job **Notifications** to send emails via Database Mail.
- Add error-checking logic in scripts (`TRY…CATCH`).
- Log job output to files or tables.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

9. Scheduling Considerations

Q: How do you decide job schedules?
A:

- Avoid peak hours for heavy operations.
- Stagger jobs to prevent resource contention.
- Coordinate with development and business teams.

10. Alternatives to SQL Server Agent

Q: What if SQL Server Agent is not available?
A:

- Use **Windows Task Scheduler** to run PowerShell or batch scripts.
- Use **Azure Automation** for cloud-based databases.
- Use **third-party schedulers** like Control-M, Jenkins.

11. Job Monitoring

Q: How do you monitor automated jobs?
A:

- Check **Job History** in SSMS.
- Query `msdb.dbo.sysjobhistory` for programmatic monitoring.
- Set up alerts for failures.

https://www.sqldbachamps.com

12. Best Practices

- Document all automated jobs.
- Test in non-production before scheduling in production.
- Implement proper error logging and alerting.
- Regularly review job execution times and adjust schedules.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

SQL Server Deployment, Patching & Upgrade – Interview Q&A

1. Deployment Process

Q: How do you support database deployments in SQL Server?
A:

- **Pre-deployment checks**: Verify DB backups, ensure sufficient disk space, check blocking jobs, and disable conflicting scheduled tasks.
- **Deployment execution**: Run scripts in a controlled order, validate objects, monitor logs.
- **Post-deployment verification**: Check schema changes, run smoke tests, and monitor performance.

2. Deployment Tools

Q: What tools do you use for deployments?
A:

- **Manual**: SQL Server Management Studio (SSMS) scripts.
- **Automated**: SQL Server Data Tools (SSDT), Redgate SQL Compare, Liquibase, or DevOps CI/CD pipelines.

3. Patch Management

Q: How do you apply SQL Server patches (CUs / Service Packs)?
A:

https://www.sqldbachamps.com

- Download patch from Microsoft site.
- Test in a non-production environment first.
- Schedule downtime or failover in an HA environment.
- Backup all databases and system DBs (`master`, `msdb`, `model`).
- Apply patch on passive node first (if in Always On or cluster), then failover and patch the other node.

4. Version Upgrade Methods

Q: What are the upgrade options in SQL Server?
A:

1. **In-place upgrade** – Replace the existing instance with the new version.
2. **Side-by-side upgrade** – Install a new SQL Server version separately, then migrate databases.
3. **Backup/Restore or Detach/Attach** – Move databases to the new instance manually.

5. Preferred Upgrade Approach

Q: Which upgrade approach do you prefer and why?
A:

- **Side-by-side** is often preferred because:

- Minimal risk to the production environment.
- Easy rollback (just switch back to old instance).
- Allows testing on the new version before cutover.

6. Pre-Upgrade Checklist

Q: What steps do you take before upgrading SQL Server?
A:

- Run **Microsoft Data Migration Assistant (DMA)** to detect deprecated features and compatibility issues.
- Check **application compatibility**.
- Ensure backups are up to date.
- Document current configurations (logins, jobs, linked servers, etc.).

7. Post-Upgrade Tasks

Q: What do you check after an upgrade?
A:

- Verify all DBs are online and accessible.
- Update database compatibility level if required.
- Recreate or migrate SQL Agent jobs, linked servers, and SSIS packages.
- Monitor performance and error logs.

https://www.sqldbachamps.com

8. Downtime Minimization

Q: How do you minimize downtime during upgrades?
A:

- Use **Always On Availability Groups** or clustering to failover.
- Use **log shipping** to keep target DBs in sync until cutover.
- Perform upgrades during off-peak hours.

9. Rollback Plan

Q: How do you prepare for rollback in case an upgrade fails?
A:

- Maintain full backups of all DBs and system databases.
- Keep old instance intact in side-by-side upgrades.
- Test rollback procedures in non-prod.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

10. Coordinating with Teams

Q: How do you coordinate deployments/upgrades with development and infrastructure teams?
A:

- Share a detailed deployment plan and schedule.
- Communicate expected downtime and risks.
- Assign clear responsibilities for each step.

11. Security During Deployments

Q: How do you ensure security is not compromised during deployment?
A:

- Restrict elevated access to authorized personnel.
- Review scripts for unauthorized changes.
- Apply security patches immediately after upgrade if needed.

12. Best Practices

- Always test deployments and patches in **non-production**.
- Maintain **version documentation** and change logs.
- Use **source control** for all deployment scripts.
- Implement **monitoring** post-deployment to catch issues early.

https://www.sqldbachamps.com

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

SQL Server Documentation & SOP – Interview Q&A

1. Importance of Documentation

Q: Why is documentation important in SQL Server administration?
A:

- Ensures **consistency** in operations.
- Reduces downtime by providing quick reference during incidents.
- Helps in **knowledge transfer** between team members.
- Supports **audit and compliance** requirements.

2. Types of Documentation

Q: What types of documentation do you maintain for SQL Server environments?
A:

1. **Configuration Documentation** – Instance details, server settings, database properties, security configurations.
2. **Process Documentation** – Step-by-step guides for common DBA tasks (backups, restores, failover).
3. **Standard Operating Procedures (SOPs)** – Formalized instructions for recurring or critical operations.
4. **Incident Reports** – Root cause analysis and resolutions.
5. **Change Logs** – Records of patches, upgrades, and configuration changes.

https://www.sqldbachamps.com

3. Configuration Documentation

Q: What details do you include in SQL Server configuration documentation?
A:

- SQL Server version, edition, and patch level.
- Hardware specs (CPU, memory, storage layout).
- Instance-level settings (max memory, MAXDOP, collation).
- Database list with sizes, recovery models, file locations.
- High Availability/Disaster Recovery (HA/DR) configurations.
- Linked servers, replication, and jobs.

4. SOP Creation

Q: How do you write a Standard Operating Procedure for SQL Server tasks?
A:

1. Define **purpose** and **scope**.
2. List **prerequisites** (access, permissions, tools).
3. Step-by-step execution instructions with screenshots if needed.

4. Error handling steps.
5. Post-execution verification.
6. Version control and review process.

5. Tools for Documentation

Q: What tools do you use to maintain SQL Server documentation?
A:

- **Word/Excel/OneNote** for manual documentation.
- **SharePoint, Confluence, or Wiki** for centralized access.
- **dbatools PowerShell module** for automated configuration exports.
- **sp_Blitz** (First Responder Kit) to auto-document server settings.

6. Automating Configuration Documentation

Q: How can you automate configuration documentation?
A:

- Use T-SQL or PowerShell scripts to extract:

```
Invoke-DbaDbServer -SqlInstance "ProdSQL01" | Export-Clixml "SQL_Config.xml"
```

- Schedule SQL Server Agent jobs to regularly export configs.

https://www.sqldbachamps.com

7. Change Tracking

Q: How do you track changes in SQL Server configuration?
A:

- Maintain a **Change Log** in version control (Git).
- Use Policy-Based Management to detect deviations.
- Schedule reports for configuration drift.

8. Documentation for Compliance

Q: How does documentation help with compliance audits?
A:

- Provides auditors with **evidence of controls**.
- Shows security roles, access reviews, backup and restore testing history.
- Demonstrates adherence to **ITIL or ISO** standards.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

9. Incident Documentation

Q: What should be documented after a SQL Server incident?
A:

- Incident timeline.
- Root cause analysis.
- Resolution steps.
- Preventive measures for the future.

10. Best Practices

- Keep documentation **up-to-date** — stale docs are as bad as no docs.
- Store in a **centralized, accessible repository**.
- Use **version control** to track updates.
- Review SOPs at least annually or after major changes.
- Include **visuals** like architecture diagrams for clarity.

SQL Server 24x7 On-Call Support – Interview Q&A

https://www.sqldbachamps.com

1. Role Understanding

Q: What does providing 24x7 on-call SQL Server support involve?
A:

- Being available to respond to **critical alerts** and incidents outside business hours.
- Monitoring and responding to issues related to **availability, performance, security, or data integrity**.
- Following **incident management procedures** to minimize downtime.
- Coordinating with infrastructure, development, and application support teams as needed.

2. Monitoring Setup

Q: How do you ensure you're alerted to SQL Server issues in real time?
A:

- Use SQL Server **alerts** (Database Mail + SQL Agent alerts for severity levels).
- Integrate with **monitoring tools** like SCOM, Redgate SQL Monitor, Idera SQL Diagnostic Manager, or SolarWinds DPA.
- Set up **PowerShell scripts** to send SMS/email alerts.
- Monitor key metrics: CPU, memory, I/O, blocking, failed backups, replication latency, Always On health.

3. Common Critical Incidents

Q: What types of incidents require immediate action in SQL Server?
A:

- SQL Server service down or cluster failover issues.
- Always On Availability Group synchronization failure.
- Blocking/deadlocks causing app outages.
- Database corruption (`DBCC CHECKDB` errors).
- Backup job failures.
- Transaction log full (especially in FULL recovery mode).

4. Incident Response Steps

Q: How do you handle a high-priority production issue?
A:

1. **Acknowledge alert** and log the incident.
2. Quickly assess the impact — is it affecting users, applications, or replication?
3. Gather diagnostics: check error logs, execution plans, `sys.dm_exec_requests`, and wait stats.
4. Apply immediate remediation (e.g., kill blocking SPID, restart failed service, switch to DR).
5. Communicate status updates to stakeholders.
6. Document the incident and schedule a post-mortem.

https://www.sqldbachamps.com

5. Minimizing Downtime

Q: How do you ensure minimal downtime during after-hours incidents?
A:

- Maintain **HA/DR** configurations (Always On, log shipping, clustering).
- Have **failover runbooks** ready.
- Keep **pre-tested recovery scripts**.
- Ensure backups are recent and verified.

6. Escalation

Q: When do you escalate a database issue during on-call?
A:

- If the incident requires **application-level changes** beyond DBA scope.
- If there's a **hardware failure** requiring infrastructure team involvement.
- If the issue is outside SQL Server control (e.g., network outage).

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

7. On-Call Preparedness

Q: How do you prepare for on-call duties?
A:

- Ensure VPN and remote access tools work.
- Keep SQL Server scripts, utilities, and documentation accessible.
- Have a list of **critical contact numbers**.
- Review the last week's incident reports to be aware of potential recurring issues.

8. Handling Database Corruption at 2 AM

Q: If you detect corruption during on-call hours, what do you do?
A:

- Run `DBCC CHECKDB` to confirm corruption.
- Check if the latest **verified backup** is available.
- Restore affected objects/databases from backup.
- If possible, failover to a healthy replica in Always On.
- Document the steps taken for the morning review.

https://www.sqldbachamps.com

9. Communication During Critical Outages

Q: How do you keep stakeholders informed during a production outage at night?
A:

- Use the **incident bridge** or chat channel.
- Provide **regular status updates** (every 15–30 minutes).
- Avoid overly technical jargon when speaking to non-technical managers.

10. Best Practices for 24x7 Support

- Automate as much monitoring as possible to reduce false alerts.
- Maintain up-to-date **runbooks** for common incidents.
- Always document resolutions to improve future response times.
- Keep a **lab/test environment** for quick reproduction and analysis.
- Practice **DR drills** regularly to keep skills sharp.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

SQL Server Internals & Performance Tuning – Interview Q&A

1. SQL Server Internals

Q: Can you explain the basic architecture of SQL Server?
A:

- **Relational Engine (Query Processor):** Handles query parsing, optimization, execution plans.
- **Storage Engine:** Manages reading/writing data to disk.
- **Buffer Manager:** Caches data pages in memory.
- **Transaction Manager:** Handles ACID compliance, logging, recovery.
- **SQLOS:** Manages memory, scheduling, and I/O for SQL Server processes.

2. Page & Extent Structure

Q: How does SQL Server store data on disk?
A:

- Data stored in **pages** (8 KB each).
- **Extents** = 8 contiguous pages (64 KB).
- Types of pages: Data, Index, GAM (Global Allocation Map), IAM (Index Allocation Map), PFS (Page Free Space).

https://www.sqldbachamps.com

3. Query Processing

Q: Describe the lifecycle of a query in SQL Server.
A:

1. **Parsing** – Query syntax checked.
2. **Binding** – Objects and columns validated.
3. **Optimization** – Execution plan created by the Query Optimizer.
4. **Execution** – Query executed against storage engine.
5. **Return Results** – Data sent to client.

4. Execution Plans

Q: How do you analyze and optimize execution plans?
A:

- Use `SET STATISTICS IO/TIME` for logical/physical reads.
- Identify scans vs. seeks.
- Watch for **key lookups**, high-cost operators (hash match, sort).
- Check estimated vs. actual rows for cardinality issues.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

5. Indexing Strategies

Q: How do you decide between clustered and nonclustered indexes?
A:

- **Clustered Index:** For range queries, sorting, and primary key lookups.
- **Nonclustered Index:** For selective queries on non-PK columns.
- Use **included columns** to create covering indexes.
- Avoid over-indexing — balance write cost vs. read benefit.

6. Statistics

Q: Why are statistics important for query performance?
A:

- Help the optimizer estimate row counts.
- Outdated statistics cause poor plan choices.
- Keep them updated via `AUTO_UPDATE_STATISTICS` or scheduled jobs.

7. Common Performance Bottlenecks

Q: What are common causes of slow SQL Server performance?
A:

https://www.sqldbachamps.com

- Missing/fragmented indexes.
- Blocking and deadlocks.
- Outdated statistics.
- Parameter sniffing issues.
- I/O subsystem latency.
- Memory pressure.

8. Parameter Sniffing

Q: What is parameter sniffing and how do you fix it?
A:

- SQL Server reuses execution plans based on first parameter values, which may not be optimal for all cases.
- Fixes:
  - Use `OPTION (RECOMPILE)` for fresh plan.
  - Use local variables inside SP.
  - Optimize for specific parameter value using query hints.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

9. Wait Statistics

Q: How do you use wait statistics for performance tuning?
A:

- Run `sys.dm_os_wait_stats` to find top waits.
- Common waits:
  - **PAGEIOLATCH_*** – I/O bottleneck.
  - **CXPACKET** – Parallelism issues.
  - **LCK_M_*** – Locking.
- Investigate root cause based on wait type.

10. Memory & CPU Tuning

Q: How do you tune SQL Server for memory and CPU efficiency?
A:

- Set **Max Server Memory** to avoid OS starvation.
- Monitor **Buffer Cache Hit Ratio** (should be > 90%).
- For CPU: Tune queries, adjust MAXDOP, check for excessive parallelism.

https://www.sqldbachamps.com

11. Query Store

Q: How does Query Store help in performance tuning?
A:

- Captures history of query execution plans.
- Identifies regressions when a new plan performs worse.
- Allows forcing a previous good plan.

12. Performance Tuning Tools

Q: What tools do you use for performance tuning?
A:

- **SQL Server Profiler** (deprecated but still useful for tracing).
- **Extended Events** for lightweight monitoring.
- **DMVs** like `sys.dm_exec_query_stats`, `sys.dm_exec_requests`.
- **sp_WhoIsActive** for active session monitoring.
- **Query Store** for plan history.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

13. Index Maintenance

Q: How do you handle index maintenance?
A:

- Rebuild if fragmentation > 30%.
- Reorganize if fragmentation between 5% and 30%.
- Update statistics post-maintenance.
- Use **maintenance windows** to avoid blocking production.

14. Real-World Optimization

Q: You have a slow-running query in production. What's your process?
A:

1. Get the execution plan.
2. Check index usage & statistics.
3. Look for blocking/deadlocks.
4. Review query logic for inefficiencies.
5. Apply tuning, test in lower environment.
6. Deploy with monitoring.

https://www.sqldbachamps.com

15. Best Practices

- Keep indexes lean and selective.
- Monitor queries using DMVs.
- Regularly update statistics.
- Use Query Store for historical insights.
- Avoid SELECT * in production queries.
- Review execution plans regularly for expensive queries.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

SQL Server HA/DR Solutions – Interview Q&A

1. HA vs DR

Q: What's the difference between High Availability (HA) and Disaster Recovery (DR) in SQL Server?
A:

- **HA**: Minimizes downtime and ensures availability within the same data center or region (e.g., Always On AG synchronous replicas, Failover Cluster Instances).
- **DR**: Ensures data availability and recovery in case of catastrophic site failures, typically across geographically separate locations (e.g., Always On AG asynchronous replicas, Log Shipping).

2. Always On Availability Groups

Q: What is Always On Availability Groups in SQL Server?
A:

- HA/DR solution introduced in SQL Server 2012.
- Allows a group of databases to fail over together.
- Supports **synchronous replication** for HA and **asynchronous replication** for DR.
- Enables **read-only secondaries** for reporting.

Q: What are prerequisites for Always On AG?
A:

- SQL Server Enterprise Edition (Basic AG available in Standard).
- Windows Server Failover Cluster (WSFC).
- Databases must be in **FULL recovery model**.
- Shared storage not required (unlike FCI).

https://www.sqldbachamps.com

Q: How do you monitor AG health?
A:

- Dashboard in SSMS (`Always On High Availability → Availability Groups`).
- DMVs like `sys.dm_hadr_availability_group_states`, `sys.dm_hadr_availability_replica_states`.
- Alerts for role changes, synchronization health, and failovers.

3. Log Shipping

Q: What is Log Shipping in SQL Server?
A:

- DR solution where transaction logs are backed up on primary and restored on secondary at scheduled intervals.
- Secondary can be in **Standby (read-only)** or **Restoring** mode.
- No automatic failover — manual intervention required.

Q: Advantages and disadvantages of Log Shipping?

A:

- **Advantages:** Simple to configure, works across editions, supports multiple secondaries.
- **Disadvantages:** No automatic failover, potential data loss depending on schedule.

4. Database Mirroring

Q: What is Database Mirroring?

A:

- Deprecated feature (replaced by Always On AG).
- Provides **synchronous (High Safety)** and **asynchronous (High Performance)** modes.
- Supports **automatic failover** when using a witness server in synchronous mode.

Q: Why is Database Mirroring deprecated?

A:

- Microsoft replaced it with Always On Availability Groups for more features, multi-database failover, and better scalability.

5. Failover Cluster Instance (FCI)

Q: What is SQL Server Failover Cluster Instance?

A:

- HA solution at the instance level using WSFC.
- Requires **shared storage** (SAN).
- Entire SQL instance (all databases) fail over together.
- Not a DR solution — doesn't protect against storage failure.

https://www.sqldbachamps.com

6. Choosing the Right HA/DR Solution

Q: How do you decide which HA/DR technology to use?

A:

- **Always On AG** – Multi-database failover, read-only replicas, Enterprise edition.
- **Log Shipping** – Simple DR, works on Standard edition, tolerates longer RPO.
- **FCI** – Instance-level HA with shared storage.
- **Database Mirroring** – Legacy systems only.

7. Common HA/DR Issues

Q: What are common issues you've faced with HA/DR setups?

A:

- Network latency causing sync delays.
- Secondary replicas out of sync due to log backup chain breaks.

- Failover failures due to WSFC misconfiguration.
- Insufficient monitoring leading to unnoticed replication failures.

8. Testing HA/DR

Q: How do you test HA/DR solutions?
A:

- Perform **planned failovers** during maintenance windows.
- Simulate network failures for DR testing.
- Validate backup/restore procedures on secondaries.
- Ensure applications can reconnect after failover.

9. RPO and RTO

Q: What's RPO and RTO in context of HA/DR?
A:

- **RPO (Recovery Point Objective)** – Maximum acceptable data loss (e.g., 5 minutes of transactions).
- **RTO (Recovery Time Objective)** – Maximum acceptable downtime (e.g., 2 minutes).
- Always On synchronous replication aims for **zero data loss** (RPO=0) and fast failover (low RTO).

https://www.sqldbachamps.com

10. Best Practices

- Always keep replicas in sync with FULL recovery model.
- Regularly test failovers.
- Monitor sync health and failover readiness.
- Keep system documentation updated.
- Use a mix of **HA and DR** for full resilience.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

T-SQL, Indexing, and Execution Plan – Interview Q&A

1. T-SQL Fundamentals

Q: What is T-SQL, and how is it different from standard SQL?
A:

- **T-SQL (Transact-SQL)** is Microsoft's proprietary extension of SQL.
- Adds procedural programming features like variables, loops, error handling (`TRY...CATCH`), and system functions.
- Supports SQL Server–specific features like `OUTPUT` clause, Common Table Expressions (CTEs), and table variables.

2. Stored Procedures

Q: What are the advantages of using stored procedures?
A:

- **Performance**: Precompiled and cached execution plans.
- **Security**: Can restrict direct table access and use parameterized execution to prevent SQL injection.
- **Maintainability**: Centralized logic, easier to update than changing multiple application queries.

Q: How do you handle optional parameters in stored procedures efficiently?
A:

- Avoid `WHERE Column = @param OR @param IS NULL` (causes poor plan reuse).
- Use **dynamic SQL** for better selective query plans.
- Alternatively, use `OPTION (RECOMPILE)` for one-off plans.

https://www.sqldbachamps.com

3. Indexing Strategies

Q: What's the difference between clustered and nonclustered indexes?
A:

- **Clustered Index**: Sorts and stores data rows physically in order; one per table.
- **Nonclustered Index**: Stores only key columns + pointer to data row; multiple per table allowed.

Q: What is a covering index?
A:

- An index that includes all columns required by a query so no lookups are needed.
- Achieved using `INCLUDE` columns.

Q: When should you use filtered indexes?
A:

- For queries accessing a subset of rows (e.g., `WHERE Status = 'Active'`).
- Saves storage and improves performance for selective queries.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

4. Execution Plan Analysis

Q: How do you view and analyze execution plans?
A:

- Use **Actual Execution Plan** in SSMS (`Ctrl+M`).
- Check for:
  - **Scans vs. Seeks** – Seek is usually better.
  - **Key Lookups** – May require covering index.
  - **Expensive Operators** – Sort, Hash Match.
  - **Estimated vs. Actual Rows** – Large differences indicate stale statistics.

Q: What's the difference between an Estimated and Actual Execution Plan?
A:

- **Estimated Plan**: Generated before execution; uses statistics to predict costs.
- **Actual Plan**: Captured after execution; includes real row counts and run time.

5. Query Optimization Techniques

Q: How do you optimize a slow T-SQL query?
A:

https://www.sqldbachamps.com

1. Check execution plan for bottlenecks.
2. Ensure proper indexing.
3. Update statistics.
4. Avoid unnecessary columns (`SELECT *`).
5. Reduce joins and subqueries where possible.
6. Break complex queries into temp tables for better optimization.

Q: What is parameter sniffing, and how do you address it?
A:

- Occurs when SQL Server caches a plan optimized for the first parameter values, which may not suit later executions.
- Fixes: `OPTION (RECOMPILE)`, local variables, or query hints like `OPTIMIZE FOR`.

6. Common DMV Queries

Q: Which DMVs help in performance troubleshooting?
A:

- `sys.dm_exec_query_stats` – Top queries by CPU, reads, writes.
- `sys.dm_exec_sql_text` – SQL text from query handle.
- `sys.dm_exec_query_plan` – Execution plan XML.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

7. Real-World Scenario

Q: A query is performing a table scan despite an index existing. What could be the reasons?

A:

- Statistics are outdated.
- Query is not selective enough.
- Index not covering required columns.
- Implicit data type conversion preventing index usage.

8. Best Practices

- Keep indexes selective, avoid over-indexing.
- Regularly update statistics.
- Use execution plans to validate optimizations.
- Parameterize queries to avoid SQL injection.
- Use table variables cautiously — temp tables often perform better for large datasets.

https://www.sqldbachamps.com

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

SSIS, SSRS, and SSAS – Interview Q&A

1. Overview

Q: What are SSIS, SSRS, and SSAS, and how do they differ?
A:

- **SSIS (SQL Server Integration Services)** – ETL (Extract, Transform, Load) tool for moving and transforming data between systems.
- **SSRS (SQL Server Reporting Services)** – Reporting platform for creating, managing, and delivering reports in multiple formats (HTML, PDF, Excel).
- **SSAS (SQL Server Analysis Services)** – OLAP and data mining engine for analytical processing using cubes and tabular models.

2. SSIS (Integration Services)

Q: How do you design an SSIS package for ETL?
A:

1. Define source and destination connections.
2. Use Data Flow Tasks for transformations (Lookup, Merge, Derived Column, etc.).
3. Implement error handling with error outputs and logging.
4. Use configuration parameters for flexibility.

https://www.sqldbachamps.com

Q: How do you handle errors in SSIS?
A:

- Redirect error rows to separate outputs.
- Use event handlers for logging.
- Maintain retry logic for transient failures.

Q: How do you optimize SSIS package performance?
A:

- Use set-based transformations instead of row-by-row operations.
- Minimize data type conversions.
- Enable **Fast Load** for destination adapters.
- Use **buffer tuning** to handle large data efficiently.

3. SSRS (Reporting Services)

Q: What are the key components of SSRS architecture?
A:

- **Report Server** – Hosts and processes reports.
- **Report Manager / Web Portal** – Web interface to manage and view reports.
- **Report Designer** – Tool (in Visual Studio or Report Builder) to create reports.

Q: How do you implement report security in SSRS?
A:

- **Item-level security** – Assign roles (Browser, Publisher, Content Manager).
- **Row-level security** – Filter data based on user identity using expressions or parameters.

Q: How can you improve SSRS report performance?
A:

- Optimize SQL queries and indexes.
- Use stored procedures instead of inline queries.
- Cache reports or enable snapshot execution.

4. SSAS (Analysis Services)

Q: What's the difference between Multidimensional and Tabular models in SSAS?
A:

- **Multidimensional** – Uses cubes, measures, and dimensions; stores data in MOLAP/ROLAP/HOLAP formats.
- **Tabular** – Uses relational tables and DAX; stores data in-memory (VertiPaq engine).

Q: How do you secure SSAS cubes?
A:

- Role-based security to control dimension members (Dimension Security).
- Cell-level security for sensitive measures.

Q: How do you process SSAS objects efficiently?
A:

- Use **incremental processing** for large cubes.
- Partition large fact tables.
- Schedule processing during off-peak hours.

https://www.sqldbachamps.com

5. Integration Between SSIS, SSRS, and SSAS

Q: How do these three services work together?
A:

- **SSIS** extracts and transforms data from sources into the data warehouse.
- **SSAS** processes the warehouse into analytical models.
- **SSRS** consumes data from SSAS cubes or warehouse tables for reporting.

6. Real-World Scenario

Q: You have a daily ETL in SSIS, but SSRS reports are showing stale data. How do you troubleshoot?
A:

- Check SSIS package execution logs.
- Verify ETL success and data load times.
- Confirm SSAS cubes (if used) are processed after ETL.
- Validate SSRS data source points to updated dataset.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

7. Best Practices

- Parameterize SSIS packages for reusability.
- Use stored procedures in SSRS for better performance and maintainability.
- Partition SSAS cubes for faster processing.
- Implement version control for all ETL/reporting artifacts.

SQL Server Security, Auditing & Compliance – Interview Q&A

1. Database Security Basics

Q: What are the main security layers in SQL Server?
A:

1. **Network Security** – Restrict access via firewalls, use encrypted connections (SSL/TLS).
2. **Instance Security** – Limit SQL Server service accounts and disable unused features.
3. **Database Security** – Implement least privilege with roles and permissions.
4. **Object-Level Security** – Grant permissions only on required objects.

Q: What's the difference between SQL Server Authentication and Windows Authentication?
A:

- **Windows Authentication**: Integrated security using AD accounts; recommended for centralized management and Kerberos support.
- **SQL Authentication**: Uses separate login/password; useful for cross-platform access but less secure if passwords are weak.

http://www.sqldbachamps.com

2. Access Controls & Roles

Q: How do you implement least privilege in SQL Server?
A:

- Assign permissions to **roles**, not individual logins.
- Use **fixed server roles** (e.g., `sysadmin`, `dbcreator`) sparingly.
- Use **fixed database roles** (`db_datareader`, `db_datawriter`, `db_owner`) or custom roles for fine-grained control.

Q: How do you secure sensitive data in SQL Server?
A:

- **Transparent Data Encryption (TDE)** for at-rest encryption.
- **Always Encrypted** for client-side encryption of sensitive columns.
- **Dynamic Data Masking (DDM)** to hide sensitive data in query results.
- Row-level security for per-user filtering.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

3. Auditing

Q: What are some SQL Server auditing options?
A:

1. **SQL Server Audit** – Native feature to track actions at the server or database level.
2. **Change Data Capture (CDC)** – Tracks DML changes for tables.
3. **C2 Audit Mode** – Legacy audit, not recommended for new implementations.
4. **Custom Triggers & Logging** – For application-specific audits.

Q: How do you implement SQL Server Audit?
A:

1. Create an **Audit** object (`CREATE SERVER AUDIT`).
2. Create an **Audit Specification** (`SERVER AUDIT SPECIFICATION` or `DATABASE AUDIT SPECIFICATION`).
3. Enable both audit and specification.
4. Review audit logs via sys.dm_audit_actions or file output.

4. Compliance Requirements

Q: How does SQL Server support compliance with standards like GDPR, HIPAA, and SOX?
A:

- **Encryption** (TDE, Always Encrypted) to protect data privacy.
- **Auditing** to maintain activity logs.
- **Row-Level Security** for data access control.
- **Data Masking** to protect PII from unauthorized view.
- **Backups encryption** for secure storage.

https://www.sqldbachamps.com

Q: How do you ensure compliance for data retention and access policies?
A:

- Implement role-based access control.
- Archive old data according to policy.
- Maintain audit logs for required retention period.
- Regularly review permissions and remove unused accounts.

5. Real-World Scenarios

Q: A compliance audit flagged excessive `db_owner` assignments. How do you fix it?
A:

- Review all `db_owner` members.
- Remove and replace with custom roles granting only needed privileges.
- Document changes for audit trail.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

Q: You suspect unauthorized access to sensitive data. How do you investigate?

A:

1. Review **SQL Server Audit** logs for SELECT operations on sensitive tables.
2. Check **login history** using `sys.dm_exec_sessions` and `sys.dm_exec_connections`.
3. Correlate events with AD security logs.

6. Best Practices

- Use **Windows Authentication** wherever possible.
- Apply **least privilege** principle to all accounts.
- Regularly rotate passwords for SQL logins.
- Encrypt sensitive data both in transit and at rest.
- Implement **auditing** for both security events and DML changes.
- Review and test compliance controls periodically.

https://www.sqldbachamps.com

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

SQL Server DBA – DevOps Tools & Practices Interview Q&A

1. General DevOps Understanding

Q: How does DevOps apply to database administration?
A:

- DevOps in DBAs focuses on **automation**, **collaboration**, and **continuous delivery** for database changes.
- It includes:
  - Version controlling database scripts with Git.
  - Automating builds and deployments via CI/CD pipelines.
  - Integrating DB change management into software release cycles.
  - Using monitoring/alerting tools for proactive support.

2. Git for Database Management

Q: How do you use Git for SQL Server database changes?
A:

- Store **T-SQL scripts**, **stored procedures**, **views**, and **schema changes** in a Git repository.
- Use **branching strategies** (feature, release, hotfix) to manage changes.
- Perform **code reviews** via pull requests.
- Integrate schema comparison tools (like Redgate SQL Source Control or SSDT) for version tracking.

https://www.sqldbachamps.com

Q: How do you handle conflicts when two developers modify the same stored procedure?
A:

- Pull latest changes from the branch before editing.
- Use Git's diff/merge tools to manually resolve conflicts.
- Test merged code in a staging database before committing.

3. CI/CD for Databases

Q: How does CI/CD work for SQL Server?
A:

1. **Continuous Integration (CI)** – Automatically build and validate DB scripts when changes are pushed to Git.
2. **Continuous Delivery/Deployment (CD)** – Automatically deploy validated changes to staging or production environments using pipeline tools (Azure DevOps, Jenkins, GitLab CI).
3. **Rollback plans** – Include scripts to reverse changes if needed.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

Q: What tools can you use for database CI/CD?
A:

- **Azure DevOps Pipelines** – Integrates with Git repos for automated deployments.
- **Octopus Deploy** – Orchestrates DB and app deployments.
- **Redgate SQL Change Automation** – Specialized for DB changes.
- **Flyway / Liquibase** – Open-source DB migration tools.

4. JIRA in DBA Workflows

Q: How is JIRA used in database administration?
A:

- Track change requests for schema modifications.
- Log incidents for production issues.
- Manage DBA tasks in **Agile/Scrum boards**.
- Link Git commits and pipeline runs to JIRA issues for traceability.

Q: How do you ensure database changes align with business requirements in JIRA?
A:

- Review **JIRA tickets** for clear acceptance criteria.
- Confirm changes have **business approval** before development.
- Link JIRA tickets to Git branches and commits for auditability.

https://www.sqldbachamps.com

5. Real-World Scenarios

Q: A developer commits a breaking DB change directly to production. How do you prevent this in a DevOps setup?
A:

- Enforce **pull request reviews** in Git.
- Use **automated schema validation** in CI pipelines.
- Restrict production deployment permissions to DBA-approved users.

Q: You have a CI/CD pipeline that fails during deployment due to a missing table. How do you troubleshoot?
A:

1. Review pipeline logs to find the failing step.
2. Check Git commits for incomplete migrations.
3. Validate the target DB schema in staging.
4. Apply missing migration scripts and re-run deployment.

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

6. Best Practices

- Store **all** database objects in Git (schema, procs, functions, triggers).
- Keep migration scripts **idempotent** (safe to run multiple times).
- Automate **unit tests** for stored procedures/functions.
- Tag releases in Git for traceability.
- Maintain **rollback scripts** for critical deployments.

https://www.sqldbachamps.com