SQL DBA Interview Questions (3 Years Experience)

1. Basics & Daily Activities

Q1. What are your daily responsibilities as a DBA?

A:

- Monitor the **health and performance** of SQL Server instances.
- Check **job statuses**, **backups**, and **error logs**.
- Manage **logins, permissions, and security policies**.
- Troubleshoot issues like blocking, deadlocks, or slow queries.
- Optimize database performance and **coordinate with developers** for deployments.

Q2. Difference between SQL Server Developer, Standard, and Enterprise Editions?

A:

- **Developer Edition** → Free, full features, but **only for development/testing**.
- **Standard Edition** → Basic features, supports small to medium workloads.
- **Enterprise Edition** → Includes **advanced features** like AlwaysOn, partitioning, compression, TDE, and better scalability.

Q3. How do you monitor the health of SQL Server instances?

A:

- Use **SSMS Activity Monitor** and **built-in DMVs** for real-time monitoring.
- Set up **SQL Server alerts** for critical events.
- Check **Windows Performance Monitor** for CPU, memory, and disk usage.
- Use **third-party monitoring tools** like Redgate, SolarWinds, or SentryOne.

Q4. Difference between clustered and non-clustered indexes?

A:

- **Clustered Index** → Sorts and stores data **physically** in the table; only **one per table**.
- **Non-Clustered Index** → Stores a **pointer to data**, not the data itself; can create **multiple per table**.
- Clustered improves **range queries**, while non-clustered improves **point lookups**.

---

2. Backup & Recovery

Q5. What are different types of backups in SQL Server?

A:

- **Full Backup** → Entire database.
- **Differential Backup** → Changes since the last full backup.
- **Transaction Log Backup** → Allows **point-in-time recovery**.
- **Copy-Only Backup** → Independent, doesn't affect backup sequence.
- **Tail-Log Backup** → Final logs before restoring.
- **File/Filegroup Backup** → Useful for **very large databases**.

Q6. How do you perform a point-in-time recovery?

A:

- Restore the latest **full backup** using NORECOVERY.
- Apply the **latest differential backup** (if any).
- Restore **transaction log backups** sequentially using the STOPAT option.

Q7. Have you faced backup failures? How did you troubleshoot?

A:
Yes, I:

- Check **SQL Server error logs** and **Windows Event Viewer**.
- Verify **disk space** and **permissions**.
- Confirm backup path accessibility.
- Run RESTORE VERIFYONLY to check backup integrity.

Q8. Difference between full, differential, and transaction log backups?

A:

- **Full** → Complete database.
- **Differential** → Only changes since last full backup.
- **Transaction Log** → Captures all transactions for **point-in-time restore**.

Q9. How do you restore a database with NORECOVERY and STANDBY options?

A:

- **NORECOVERY** → Keeps the DB in **restoring mode**, allowing multiple backups to be applied.
- **STANDBY** → DB is in **read-only mode** between restores, useful for reporting servers.

---

3. Performance Tuning

Q10. What steps do you take when a query is running slow?

A:

- Check the **execution plan** for missing indexes or scans.
- Review **statistics** and **update if stale**.
- Use **DMVs** to analyze resource usage.
- Optimize indexes or rewrite queries where needed.

Q11. What are wait statistics?

A:

- Wait statistics show where **SQL Server spends time waiting**.
- Examples:
    - **PAGEIOLATCH** → Disk I/O waits.
    - **CXPACKET** → Parallelism waits.
- Used for **root cause analysis** in performance tuning.

Q12. How do you check for blocking sessions?

A:

- Use:
- SELECT * FROM sys.dm_exec_requests WHERE blocking_session_id <> 0;
- Or check **Activity Monitor** in SSMS.

Q13. Difference between SQL Profiler and Extended Events?

A:

- **SQL Profiler** → GUI-based, heavier, and older.
- **Extended Events** → Lightweight, modern, and better for **production monitoring**.

Q14. What are DMVs? Which do you use for performance tuning?

A:
DMVs (**Dynamic Management Views**) give **real-time insights** into SQL Server.

- Examples I use:
    - sys.dm_exec_query_stats → Expensive queries.
    - sys.dm_exec_requests → Running queries.
    - sys.dm_os_wait_stats → Wait analysis.

---

4. High Availability & Disaster Recovery (HADR)

Q15. Difference between Log Shipping, Mirroring, AlwaysOn, and Clustering?

A:

- **Log Shipping** → Periodic log backups shipped to secondary.
- **Mirroring** → Real-time single database replication.
- **AlwaysOn AG** → Multi-database HA/DR with automatic failover.
- **Clustering** → OS-level failover using shared storage.

Q16. How do you configure and monitor AlwaysOn Availability Groups?

A:

- Enable **AlwaysOn** in SQL Server configuration.
- Create an **Availability Group** and add replicas.
- Set **failover modes** and configure **endpoints**.
- Monitor via **Availability Dashboard** and **DMVs**.

Q17. Advantages and disadvantages of Log Shipping?

A:

- **Advantages** → Simple, cost-effective, reliable.
- **Disadvantages** → Manual failover, possible data delay.

Q18. What happens if the primary node in AlwaysOn goes down?

A:

- If **automatic failover** is enabled, a secondary replica becomes primary.

- Otherwise, manual failover is required.

---

5. Security & Maintenance

Q19. How do you handle logins, users, and permissions?

A:

- Create logins at the server level.
- Map logins to **database users**.
- Assign **roles** or specific permissions using **GRANT**, **DENY**, and **REVOKE**.
- Regularly audit permissions for compliance.

Q20. Difference between SQL Server Authentication and Windows Authentication?

A:

- **Windows Authentication** → Uses AD credentials, more secure.
- **SQL Authentication** → Separate SQL logins, useful when AD is unavailable.

Q21. How do you implement row-level security?

A:

- Create a **security predicate** and **SECURITY POLICY**.
- Filters data based on **user roles** or login context.

Q22. How do you audit login failures in SQL Server?

A:

- Enable login auditing in **SQL Server Properties**.
- Check **SQL error logs** and **sys.fn_get_audit_file**.

Q23. What is Transparent Data Encryption (TDE)?

A:

- TDE encrypts database files (**.mdf, .ldf**) at rest.
- Protects data even if someone **steals database files**.

---

6. Troubleshooting & Real-time Scenarios

Q24. What will you do if the transaction log file grows very large?

A:

- Check for **open transactions** using DBCC OPENTRAN().
- Take **log backups** to free space.
- Switch recovery model to **SIMPLE** if log backups are not needed.

Q25. How do you handle blocking and deadlocks?

A:

- Use DMVs to find **blocking sessions**.
- Capture **deadlock graphs** using Extended Events.
- Optimize queries and indexes to prevent locks.

Q26. If a database goes into suspect mode, how do you recover it?

A:

- Set database to **EMERGENCY mode**:
- ALTER DATABASE DBName SET EMERGENCY;
- Run DBCC CHECKDB to detect corruption.
- Restore from **latest backup** if needed.

Q27. How do you monitor disk space and memory usage for SQL Server?

A:

- Use **sys.dm_os_performance_counters** and SSMS reports.
- For enterprise monitoring, I prefer tools like **Redgate** or **SolarWinds**.

Q28. Have you faced high CPU utilization issues? How did you fix them?

A:

- Use sys.dm_exec_query_stats to identify **high-cost queries**.
- Tune **execution plans** and add **missing indexes**.
- Optimize queries or move workloads to off-peak hours.

---

7. SQL Queries (Hands-On)

Q29. Find the second highest salary from an employee table.

```
SELECT MAX(salary)
FROM employees
WHERE salary < (SELECT MAX(salary) FROM employees);
```

Q30. Difference between DELETE, TRUNCATE, and DROP.

- **DELETE** → Removes specific rows, can use WHERE, fully logged.
- **TRUNCATE** → Removes all rows, minimal logging, faster.
- **DROP** → Deletes table structure completely.

Q31. Find duplicate rows in a table.

```
SELECT column1, COUNT(*)
FROM table_name
GROUP BY column1
HAVING COUNT(*) > 1;
```

Q32. Explain CTEs, temp tables, and table variables.

- **CTEs** → Simplify **complex queries** and **recursive queries**.
- **Temp Tables** → Stored in tempdb, best for **large datasets**.
- **Table Variables** → Stored in memory, best for **small datasets**.

Q33. How do you use partitioning in SQL Server?

A:

- Partitioning splits **large tables** into smaller pieces.
- Improves **query performance** and **data maintenance**.