■ Application Developer Interview Prep (Python + Databricks)

■ Technical Questions (Python & Databricks)

Q1. Can you explain Python's memory management and garbage collection mechanism?
A: Python uses reference counting and a garbage collector to manage memory automatically.

Q2. How would you optimize a Python script that processes millions of records daily?
A: Use efficient data structures, vectorization (NumPy/Pandas), multiprocessing, or chunked processing.

Q3. What are Python decorators and when would you use them?
A: Functions that modify other functions; useful for logging, authentication, caching.

Q4. Can you explain the difference between shallow copy and deep copy in Python?
A: Shallow copy copies references, deep copy creates completely new objects recursively.

Q5. Difference between @classmethod, @staticmethod, and instance methods?
A: classmethod works on class, staticmethod doesn't need class/instance, instance methods work on object.

Q6. How do you handle exceptions in Python?
A: Using try-except-finally; can define custom exception classes.

Q7. What is the Global Interpreter Lock (GIL) in Python?
A: Lock that prevents multiple threads from executing Python bytecode simultaneously.

Q8. Explain generators and iterators in Python with real-world use cases.
A: Iterators implement __iter__ and __next__; Generators use 'yield' for memory-efficient data streams.

Q9. How would you integrate Databricks with Python for data processing?
A: Using Databricks Connect, SparkSession, or APIs in Python.

Q10. Difference between Databricks Notebooks, Jobs, and Workflows?
A: Notebook = interactive dev, Job = scheduled task, Workflow = orchestration of jobs.

Q11. How do you manage data pipelines in Databricks?
A: Using Delta Live Tables, Workflows, and APIs.

Q12. Can you explain how Delta Lake works in Databricks?
A: Delta Lake provides ACID transactions and schema enforcement on data lakes.

Q13. How do you handle schema evolution in Databricks tables?
A: Enable schema evolution during writes or ALTER TABLE operations.

Q14. How would you optimize Spark jobs on Databricks?
A: Optimize partitioning, caching, broadcast joins, cluster configuration.

Q15. Difference between RDD, DataFrame, and Dataset?
A: RDD = low-level, DataFrame = structured API, Dataset = typed API (Scala/Java).

■ Software Development Practices & Collaboration

Q16. Can you explain Agile methodology?
A: Iterative dev cycle with sprints, daily stand-ups, user stories.

Q17. How do you ensure code quality in teams?
A: Code reviews, unit tests, linting, CI/CD pipelines.

Q18. How do you handle version control conflicts in Git?
A: Pull latest changes, resolve conflicts manually, commit merged code.

Q19. Approach to writing unit tests in Python?
A: Use unittest or pytest, write small independent tests.

Q20. Explain CI/CD implementation experience.
A: Automated build, test, deployment using tools like Jenkins/GitHub Actions.

■ Problem-Solving & Debugging

Q21. Debugging a complex production issue?
A: Reproduce locally, analyze logs/metrics, isolate root cause, fix & test.

Q22. How do you analyze logs/metrics?
A: Using log analyzers, monitoring tools, error patterns.

Q23. Databricks job failing due to memory errors?
A: Optimize partition size, caching, increase cluster memory, avoid wide transformations.

Q24. How to profile and optimize slow Python code?
A: Use cProfile, line_profiler, optimize hotspots, improve algorithms.

■ Collaboration & SME Role

Q25. Have you mentored juniors?
A: Yes, through code reviews, pair programming, training sessions.

Q26. Contribution to team discussions?
A: Share solutions, evaluate trade-offs, align with business needs.

Q27. Handling conflicting solutions?
A: Compare pros/cons, run small POC, align with team consensus.

Q28. Ensuring client requirements → technical solutions?
A: Gather requirements, write technical specs, confirm with client.

Q29. Example of process improvement?
A: Suggested automation scripts that reduced manual work/time.

■ Behavioral & Situational (MR-type)

Q30. What motivates you as SDE?
A: Solving complex problems, building scalable apps, continuous learning.

Q31. Handling tight deadlines?
A: Prioritize tasks, communicate risks, deliver MVP first.

Q32. Disagreed with senior's approach?
A: Discuss respectfully, provide alternatives, accept final decision.

Q33. Example of cross-functional teamwork?
A: Worked with QA, DevOps, Analysts to deliver end-to-end features.

Q34. Balance clean code vs deadlines?
A: Write clean modular code, refactor later if urgent.

Q35. What if stuck on a problem too long?
A: Research, ask peers, escalate if blocking, document learning.