(continued from previous section...)

Basic Technical Questions by Technology (Continued)

Node.js & TypeScript

63. **What is middleware in Express.js?**

Middleware functions in Express.js are functions that have access to the request and response objects. They can modify the request, execute logic, and end the request-response cycle or call the next middleware in the stack.

64. **How do you handle asynchronous code in Node.js?**

Use Promises or async/await for cleaner, non-blocking code. Node.js also provides callback patterns but modern code prefers Promises/async-await.

65. **What are some common types in TypeScript?**

Primitive types (string, number, boolean), arrays, tuples, enums, interfaces, union and intersection types, and any/unknown/never.

PostgreSQL & TimescaleDB

66. **How do you create and use a view in PostgreSQL?**

A view is a saved SQL query that acts like a virtual table. You create it using `CREATE VIEW view_name AS SELECT ...`. It simplifies complex queries and can encapsulate logic.

67. **What is query planning in PostgreSQL?**

PostgreSQL uses a query planner to determine the most efficient way to execute a SQL query. You can inspect plans using `EXPLAIN` or `EXPLAIN ANALYZE`.

68. **How does TimescaleDB handle high-ingestion rates?**

TimescaleDB uses hypertables and partitioning to write and query time-series data efficiently, scaling to millions of inserts per second.

GraphQL & Apollo Server

69. **What is a schema in GraphQL?**

A schema defines the structure of data that can be queried via a GraphQL API. It includes types, queries, mutations, and optionally subscriptions.

70. **What are GraphQL mutations?**

Mutations are used to modify data on the server (similar to POST, PUT, DELETE in REST). They return a result and can trigger side effects.

Microservices Architecture

71. **What are some common challenges with microservices?**

Challenges include data consistency, inter-service communication, observability, deployment complexity, and handling failures across services.

72. **What is a bounded context?**

A bounded context is a core concept in Domain-Driven Design (DDD) that defines clear boundaries within which a specific model applies. It's used to divide systems into cohesive, decoupled services.

Docker & Kubernetes

73. **What is a Dockerfile?**

A Dockerfile is a text file containing instructions for building a Docker image, including base image, copied files, environment variables, and command to run.

74. **What is a Kubernetes Deployment?**

A Deployment manages a set of replicated Pods, provides declarative updates, and ensures availability by managing rollouts and rollbacks.

Kafka / RabbitMQ / NATS / Redis Streams

75. **What is a Kafka topic?**

A topic in Kafka is a named stream of records. Producers write to topics, and consumers read from them. Topics are partitioned for scalability.

76. **How does Redis Streams differ from Pub/Sub?**

Redis Streams supports persistence, message acknowledgment, and replay, whereas Pub/Sub is ephemeral and doesnt persist messages.

Redis

77. **What is a Redis hash?**

A Redis hash maps fields to values, like a dictionary. Its efficient for storing related data, such as user objects.

78. **How do you implement distributed locks in Redis?**

Use the `SET key value NX PX timeout` command or libraries like Redlock to ensure mutual exclusion in distributed systems.

CI/CD

79. **What are GitHub Actions?**

GitHub Actions is a CI/CD tool that runs workflows defined in YAML files triggered by events like push, pull request, or schedule.

80. **What is ArgoCD used for?**

ArgoCD is a GitOps continuous delivery tool for Kubernetes. It synchronizes Kubernetes clusters with declarative configurations stored in Git.

Monitoring & Observability

81. **What is the difference between logs and metrics?**

Logs are event-based records (e.g., errors, requests), while metrics are numeric measurements collected over time (e.g., CPU usage, request latency).

82. **What is tracing and why is it important?**

Tracing shows the lifecycle of a request across services. It helps identify bottlenecks and failures in distributed systems.

This comprehensive list of basic and advanced questions and answers should prepare you well for the Backend Developer role at Pragathi Solutions.