1. **MongoDB বা MySQL** বেছে নেওয়ার সময় নিচের বিষয়গুলো বিবেচনা করতে পারেন:

MongoDB নির্বাচন করবেন যখন:
- ডেটা স্ট্রাকচার অনিয়মিত (Unstructured বা Semi-Structured):
- স্কেলেবিলিটি দরকার:
  MongoDB সহজে horizontal scaling সাপোর্ট করে। যদি আপনার অ্যাপ ভবিষ্যতে বড় পরিসরে ডেটা হ্যান্ডেল করবে, MongoDB উপযুক্ত।
- ফাস্ট ডেভেলপমেন্ট চাচ্ছেন:
- রিয়েল-টাইম ডেটা প্রসেসিং: উদাহরণ: লাইভ চ্যাট অ্যাপ।
- জিওগ্রাফিক ডেটা:

MySQL নির্বাচন করবেন যখন:

- স্ট্রাকচারড ডেটা (Structured Data):
- ট্রানজেকশন দরকার (ACID Properties):
- কমপ্লেক্স কোয়েরি দরকার:
- পরিপক্কতা এবং কমিউনিটি সাপোর্ট:
- রিপোর্টিং এবং ডেটা অ্যানালাইসিস:

উদাহরণ:

- **MongoDB:**
  সোশ্যাল মিডিয়া অ্যাপ, ই-কমার্স প্রোডাক্ট লিস্টিং, চ্যাট অ্যাপ।

- **MySQL:**
  ব্যাংকিং সিস্টেম, হিসাবরক্ষণ সফটওয়্যার, স্টুডেন্ট ম্যানেজমেন্ট সিস্টেম।

উপসংহার:
আপনার অ্যাপ্লিকেশনের প্রয়োজন অনুযায়ী ডেটাবেস নির্বাচন করুন। যদি ডেটা রিলেশনাল হয়, তবে MySQL। আর যদি ফ্লেক্সিবিলিটি এবং স্কেলেবিলিটি দরকার হয়, তবে MongoDB।

ক্লোজার (Closure) হলো একটি ফাংশন যা তার আউটার স্কোপের ভেরিয়েবলগুলোকে অ্যাক্সেস করতে পারে, এমনকি যখন আউটার ফাংশনটি এক্সিকিউটেড হয়ে যায় বা এর এক্সিকিউশন শেষ হয়ে যায়।

আরেকটু সহজভাবে বললে, ক্লোজার একটি ফাংশন এবং তার আউটার স্কোপের ভেরিয়েবলগুলোর সাথে সম্পর্কিত একটি রেফারেন্সের সমন্বয়।

হায়ার-অর্ডার ফাংশনের গুণাবলী:

1. **এক বা একাধিক ফাংশনকে আর্গুমেন্ট হিসেবে নিতে পারে।**
2. **ফাংশনকে রিটার্ন (ফিরিয়ে) করতে পারে।**

Summary: Promise vs Async Await

Feature	`Promise`	`async/await`
Syntax	`.then()`, `.catch()` chaining	`async function` and `await` keyword
Error Handling	`.catch()`	`try/catch` block
Readability	Somewhat less readable (due to chaining)	More readable (like synchronous code)
Return Value	Returns a `Promise` object	Returns a `Promise`, but can be used like synchronous code
Chaining	Requires chaining `.then()`	No chaining required

Conclusion:

- `Promise` দিয়ে asynchronous কোড লেখা যেতে পারে, তবে তা কিছুটা complex হয়ে যেতে পারে।
- `async/await` কোডটিকে সহজ, পরিষ্কার এবং synchronous কোডের মতো করে তোলে। তবে, এটি `Promise` এর উপর ভিত্তি করে কাজ করে।

যেহেতু `async/await` কে মনে হবে আরো পরিষ্কার এবং readable, তাই এটি বর্তমানের ভালো পদ্ধতি হিসাবে ব্যবহৃত হয়। তবে, `Promise` এর ধারণা জানাও গুরুত্বপূর্ণ, কারণ `async/await` এর পিছনে `Promise` কাজ করে।