✅ Pros:

Full control — Define datatypes, sizes, constraints, defaults. You're the architect here.

Clear structure — Makes code easier to read, maintain, and debug.

Can be extended — You can add indexes, constraints, and even do funky things like computed columns or check constraints.

❌ Cons:

Takes longer to write — More verbose, more keystrokes.

Overkill for simple one-offs — Feels like bringing a sledgehammer to crack a nut when you're just grabbing throwaway data.

🧠 When to Use Which?

|  |  |
| --- | --- |
| Scenario | Use This |
| Quick troubleshooting / ad hoc | SELECT INTO |
| Building something reusable or production-grade | CREATE TABLE + INSERT |
| Need to define column types explicitly | CREATE TABLE + INSERT |
| Working with dynamic SQL and don’t want to declare schema upfront | SELECT INTO |
| Need indexes or constraints on the temp table | CREATE TABLE + INSERT |

💡 Bonus DBA Insight:

If you're using temp tables in stored procedures, always go with CREATE TABLE, because:

It plays nicer with recompilation and plan reuse.

Helps avoid temp table caching weirdness.

But if you're in script land, and you just wanna try stuff real quick before you hit the break room — fire away with SELECT INTO.