**Hybrid Cloud**

**What.** Hybrid cloud blends a private environment (on-prem or private cloud) with public-cloud services, tied together over secure networking so apps and data can live where they fit best. NIST defines cloud as on-demand, shared, configurable resources; “hybrid” is the deployment pattern that mixes environments under one architecture. Big vendors describe it as unifying on-prem, private, and public into one flexible platform.

**How.** If I take our JSP/Tomcat + MySQL project, I’d run the web tier in a public cloud (easy scaling and quick redeploys) and keep MySQL on a private network (tighter control). They talk over a VPN/peering link. Same code, different endpoints: validation and parameterized SQL don’t change; only the JDBC URL does. Ops-wise, I’d lock down inbound rules, tag resources for cost tracking, and script infra so dev/test/prod stay consistent.

// Simple hybrid toggle for JDBC endpoint

String env = System.getenv("APP\_ENV"); // LOCAL or CLOUD

String jdbc = "jdbc:mysql://localhost:3306/CSD430?useSSL=false";

if ("CLOUD".equalsIgnoreCase(env)) {

jdbc = "jdbc:mysql://10.0.1.25:3306/CSD430?useSSL=true"; // private IP via VPN/peering

}

try (var c = java.sql.DriverManager.getConnection(jdbc, "student1", "pass")) {

// UPDATE/CRUD via PreparedStatement as usual

}

**Why.** Hybrid gives agility without giving up control: bursty front ends scale fast in the public cloud, while sensitive data stays close to home and policies. Providers document common patterns (private subnets, peering, and managed services) so teams get repeatable architectures instead of snowflakes. The trade-offs are real—extra networking complexity and potential egress costs—but the flexibility usually outweighs them when you need both speed and governance.

**References**

IBM. (n.d.). *What is hybrid cloud?* <https://www.ibm.com/think/topics/hybrid-cloud>

Amazon Web Services. (n.d.). *What is a hybrid cloud?* <https://aws.amazon.com/what-is/hybrid-cloud/>