Corrigé détaillé

Terraform Associate 004 — 50 questions

Grille de réponses express, puis **explication** de chaque question.
Questions : **QUIZ-TERRAFORM-004-EXAMEN.md**

Grille express — Q1 à Q25

Q	Rép	Q	Rép	Q	Rép	Q	Rép	Q	Rép
1	B	6	B	11	B	16	C	21	C
2	B	7	A	12	B	17	A	22	B
3	B	8	B	13	B	18	C	23	B
4	C	9	B	14	B	19	B	24	B
5	C	10	A	15	C	20	B	25	B

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Grille express — Q26 à Q50

Q	Rép	Q	Rép	Q	Rép	Q	Rép	Q	Rép
26	B	31	B	36	B	41	B	46	B
27	B	32	B	37	B	42	B	47	B
28	B	33	B	38	B	43	C	48	A
29	A	34	B	39	B	44	B	49	B
30	B	35	A	40	**A + B**	45	A	50	B

Q40 attend DEUX réponses (A et B).

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Explications

Domaines 1 & 2 — Q1 à Q10

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Q1 → Q5

- **Q1 · B** — l'IaC apporte reproductibilité, versionnage (Git) et revue. Ce n'est ni du chiffrement ni un remplacement d'équipe.
- **Q2 · B** — Terraform **provisionne** (crée/gère) l'infra de façon **déclarative**. (Ansible, lui, est plutôt du *configuration management*.)
- **Q3 · B** — déclaratif = on décrit **l'état désiré**, le moteur calcule les actions. L'impératif décrirait les étapes.
- **Q4 · C** — l'agnosticisme vient des **providers** : un plugin par cloud/service.
- **Q5 · C** — la configuration s'écrit en **HCL** (le JSON est aussi accepté, mais HCL est la forme native).

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Q6 → Q10

- **Q6 · B** — `init` prépare le répertoire : **télécharge les providers**, configure le backend, installe les modules. Idempotent.
- **Q7 · A** — `required_providers` (dans `terraform {}`) déclare **source** + **contrainte de version** de chaque provider.
- **Q8 · B** — `~> 1.2.0` = opérateur pessimiste à 3 segments → **`>= 1.2.0` et `< 1.3.0`** (seul le dernier nombre peut augmenter).
- **Q9 · B** — le bloc `provider` **configure** le provider : région, souscription, auth, options (`features {}` pour azurerm).
- **Q10 · A** — `required_version` épingle la version du **binaire Terraform** ; `version` (dans `required_providers`) concerne les providers.

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Explications

Domaine 3 — Core workflow — Q11 à Q18

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Q11 → Q14

- **Q11 · B** — workflow de base : `init` → `plan` → `apply` (puis `destroy` pour nettoyer).
- **Q12 · B** — `plan` **prévisualise** le diff (`+ ~ -`) sans rien changer. `show` lit un plan/état existant.
- **Q13 · B** — `fmt` **reformate** le code selon le style canonique (indentation, alignement).
- **Q14 · B** — `validate` vérifie la **validité interne** de la config (syntaxe, références, types) **sans appeler** les API cloud.

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Q15 → Q18

- **Q15 · C** — `terraform destroy` détruit tout le périmètre géré.
- **Q16 · C** — `-auto-approve` saute la confirmation (à réserver à la **CI**).
- **Q17 · A** — sauvegarder le plan (`-out`) puis `apply tf.plan` applique **exactement** ce qui a été relu, sans recalcul ni nouvelle confirmation.
- **Q18 · C** — `-/+` = **replace** : Terraform **détruit puis recrée**. Cherchez `# forces replacement`. (`~` = update in-place, sans danger.)

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Explications

Domaine 4 — Configuration — Q19 à Q32

Q19 → Q23

- **Q19 · B** — une variable d'entrée se déclare avec `variable "nom" { ... }`.
- **Q20 · B** — on la référence via **`var.region`**.
- **Q21 · C** — ordre de précédence (le plus fort en dernier appliqué) : `default` < env `TF_VAR_*` < `terraform.tfvars` < `*.auto.tfvars` < **`-var` / `-var-file` (CLI)**.
- **Q22 · B** — `output` **expose** une valeur (affichée après l'apply, lisible par d'autres modules).
- **Q23 · B** — `sensitive = true` **masque** la valeur dans la sortie CLI et les plans. ⚠️ Elle reste **en clair dans le state**.

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Q24 → Q28

- **Q24 · B** — une **data source LIT** l'existant (que des sorties) ; une **ressource CRÉE/gère**.
- **Q25 · B** — on préfixe par `data.` : **`data.azurerm_dns_zone.labs.name`**.
- **Q26 · B** — `locals` = **valeurs locales** calculées, réutilisables (`local.nom`), pour éviter la répétition.
- **Q27 · B** — `count = N` crée **N instances** (indexées `count.index`).
- **Q28 · B** — `for_each` (map/set) donne des **clés stables** : retirer une entrée ne recrée pas les autres (contrairement à `count`, indexé).

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Q29 → Q32

- **Q29 · A** — `depends_on` force une dépendance **explicite** quand elle n'est pas déductible d'une référence d'attribut.
- **Q30 · B** — `prevent_destroy = true` fait **échouer** tout apply qui détruirait la ressource. Coupe-circuit anti-erreur.
- **Q31 · B** — `create_before_destroy = true` crée le **remplaçant avant** de supprimer l'ancien → évite la coupure de service.
- **Q32 · B** — les **provisioners** (`remote-exec`, `local-exec`…) sont un **dernier recours** : on privilégie les ressources/providers natifs ou cloud-init.

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Explications

Domaines 5 & 6 — Q33 à Q44

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Q33 → Q37

- **Q33 · B** — un **module** est un conteneur réutilisable : `variable` en entrée, `output` en sortie. Une « fonction » d'infra.
- **Q34 · B** — le **module racine** est le répertoire de travail courant (là où on lance `terraform`).
- **Q35 · A** — on passe les valeurs via les **arguments** du bloc `module` (= ses variables d'entrée).
- **Q36 · B** — on lit une sortie via **`module.vm.ip`**.
- **Q37 · B** — `source = "Azure/naming/azurerm"` référence un module du **Terraform Registry** au format `namespace/nom/provider`.

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Q38 → Q44

- **Q38 · B** — le **state** mappe la config aux **ressources réelles** (la mémoire de Terraform).
- **Q39 · B** — par défaut, state **local** dans `terraform.tfstate`.
- **Q40 · A + B** — un backend distant apporte le **partage** d'équipe **et** le **verrouillage**. Il n'accélère pas l'apply et ne supprime pas les providers.
- **Q41 · B** — sur **Azure Storage**, verrou = **blob lease** (natif). DynamoDB, c'est l'ancien duo côté AWS/S3.
- **Q42 · B** — `terraform state list` liste les ressources gérées.
- **Q43 · C** — `terraform state mv` **réétiquette** la ressource dans le state → le plan affiche « No changes ».
- **Q44 · B** — Terraform **rafraîchit** le state au `plan` et **affiche la dérive** par rapport au réel.

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Explications

Domaines 7 & 8 — Q45 à Q50

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Q45 → Q48

- **Q45 · A** — `terraform import` intègre une ressource **existante** dans le state (puis on écrit le bloc `resource`). Terraform 1.5+ permet aussi des blocs `import {}` déclaratifs.
- **Q46 · B** — `terraform apply -replace=<addr>` remplace l'ancien `terraform taint` (déprécié) pour **forcer la recréation**.
- **Q47 · B** — `terraform state rm` **oublie** la ressource (retirée du state) **sans la détruire** réellement dans le cloud.
- **Q48 · A** — les **workspaces** CLI permettent **plusieurs états** à partir de la **même configuration** (ex. dev/staging).

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Q49 → Q50

- **Q49 · B** — **HCP Terraform** (ex-Terraform Cloud) : exécution **distante**, **state managé** + verrou, **RBAC**, VCS, **policies** (Sentinel/OPA), health assessments.
- **Q50 · B** — les **variable sets** permettent de **partager un même jeu de variables** (ex. credentials, région) entre **plusieurs workspaces / projets**, sans les redéfinir à chaque fois.

© 2026 — Formation Terraform / Azure · Questions : QUIZ-TERRAFORM-004-EXAMEN.md

Calcul de votre score 🎯

- **≥ 35 / 50 (70 %)** → niveau visé pour l'examen 004
- **45–50 / 50** → très solide, vous êtes prêt
- Revoyez en priorité les domaines où vous avez < 60 %

Réf. officielle : developer.hashicorp.com/certifications/infrastructure-automation