✔ Tests SQL commentés — Trouve ton artisan

Ce document reprend les requêtes du script `sql/04_tests.sql`. Pour chaque test :

- objectif,
- requête SQL,
- résultat attendu,
- résultat obtenu (preuves),
- commentaire.

💠 Test 01 — Contrôle de volumétrie (sanity check)

But : vérifier que les tables principales sont bien peuplées après l'exécution du seed.

Code SQL :

```
SELECT 'categories' AS table_name, COUNT(*) AS nb FROM categories
UNION ALL
SELECT 'specialties', COUNT(*) FROM specialties
UNION ALL
SELECT 'artisans', COUNT(*) FROM artisans;
```

Résultat attendu : des compteurs cohérents avec le fichier Excel (aucune table à 0). Résultat obtenu (preuve) :

- Export PDF : tests/exports/test01.pdf

Commentaire : si artisans = 0, le seed n'a pas été exécuté ou a échoué.

💠 Test 02 — Menu : liste des catégories (ordre alphabétique)

But : valider les données destinées au menu du header (catégories).

Code SQL :

```
SELECT id, name
FROM categories
ORDER BY name ASC;
```

Résultat attendu : 4 catégories triées A→Z. Résultat obtenu (preuve) :

- Export PDF : tests/exports/test02.pdf

Commentaire : correspond directement à la donnée affichée dans la navigation du site.

💠 Test 03 — Spécialités rattachées à une catégorie

But : vérifier la cohérence de la relation catégories → spécialités.

Code SQL :

```
SELECT
  c.name AS category,
  COUNT(*) AS nb_specialties
FROM specialties s
JOIN categories c ON c.id = s.category_id
GROUP BY c.id, c.name
ORDER BY c.name;
```

Résultat attendu : chaque spécialité est liée à une catégorie ; le compteur est cohérent. Résultat obtenu (preuve) :

- Export PDF : tests/exports/test03.pdf

Commentaire : si une catégorie a 0 spécialité, cela peut impacter l'affichage côté front.

🧩 Test 04 — Liste des artisans par catégorie (page catégorie)

But : simuler la requête utilisée sur la page de liste filtrée par catégorie.

Code SQL :

```
SELECT
  a.id,
  a.name,
  a.rating,
  s.name AS specialty,
  a.city
FROM artisans a
JOIN specialties s ON s.id = a.specialty_id
JOIN categories c ON c.id = s.category_id
WHERE c.name = 'Bâtiment'
ORDER BY a.rating DESC, a.name ASC;
```

Résultat attendu : une liste d'artisans appartenant à la catégorie demandée (ici la catégorie: Bâtiment)
Résultat obtenu (preuve) :

- Export PDF : tests/exports/test04.pdf

Commentaire : cette requête servira à l'API (ex: GET /categories/:id/artisans).

🧩 Test 05 — Recherche (barre de recherche)

But : valider la recherche utilisée sur le site via un mot-clé.

Code SQL :

```
SET @q = 'boul';

SELECT
  a.id,
  a.name,
  a.rating,
  s.name AS specialty,
  a.city,
  c.name AS category
FROM artisans a
JOIN specialties s ON s.id = a.specialty_id
JOIN categories c ON c.id = s.category_id
WHERE a.name LIKE CONCAT('%', @q, '%')
   OR s.name LIKE CONCAT('%', @q, '%')
   OR a.city LIKE CONCAT('%', @q, '%')
ORDER BY a.rating DESC, a.name ASC;
```

Résultat attendu : des résultats pertinents selon le mot-clé choisi (ici boul). Résultat obtenu (preuve) :

- Export PDF : tests/exports/test05.png

Commentaire : le brief impose la recherche sur le nom ; ici la recherche est étendue (bonus) à la ville et spécialité.

Test 06 — Accueil : artisans du mois (3 maximum)

But : récupérer les artisans “Top” mis en avant sur la page d’accueil.

Code SQL :

```
SELECT
  a.id,
  a.name,
  a.rating,
  s.name AS specialty,
  a.city
FROM artisans a
JOIN specialties s ON s.id = a.specialty_id
WHERE a.is_featured = 1
ORDER BY a.rating DESC, a.name ASC
LIMIT 3;
```

Résultat attendu : 3 lignes maximum, uniquement is_featured = 1. Résultat obtenu (preuve) :

- Export PDF : tests/exports/test06.pdf

Commentaire : si moins de 3 résultats, le dataset contient moins de 3 artisans “Top”.

Test 07 — Fiche artisan : récupération complète

But : vérifier que toutes les informations de la fiche artisan sont récupérables.

Code SQL :

```
SET @artisan_id = 1;

SELECT
  a.id,
  a.name,
  a.rating,
  a.city,
  a.about,
  a.email,
  a.website,
  a.image_url,
  a.is_featured,
  s.name AS specialty,
  c.name AS category
FROM artisans a
JOIN specialties s ON s.id = a.specialty_id
JOIN categories c ON c.id = s.category_id
WHERE a.id = @artisan_id;
```

Résultat attendu : 1 ligne complète correspondant à l'artisan demandé (ici l'artisan ayant pour id: 1). Résultat obtenu (preuve) :

- Export PDF : tests/exports/test07.png

Commentaire : cette requête correspond à l'endpoint de fiche (ex: GET /artisans/:id).

Test 08 — Contrôle qualité : artisans sans spécialité

But : vérifier l'intégrité relationnelle (doit être 0).

Code SQL :

```
SELECT COUNT(*) AS artisans_without_specialty
FROM artisans a
LEFT JOIN specialties s ON s.id = a.specialty_id
WHERE s.id IS NULL;
```

Résultat attendu : 0 Résultat obtenu (preuve) :

- Export PDF : tests/exports/test08.pdf

Commentaire : si >0, il y a une incohérence seed/relations.

🧩 Test 09 — Contrôle qualité : spécialités sans catégorie

But : vérifier l'intégrité relationnelle (doit être 0).

Code SQL :

```
SELECT COUNT(*) AS specialties_without_category
FROM specialties s
LEFT JOIN categories c ON c.id = s.category_id
WHERE c.id IS NULL;
```

Résultat attendu : 0 Résultat obtenu (preuve) :

- Export PDF : tests/exports/test09.pdf

Commentaire : si >0, une spécialité n'est pas correctement rattachée.