Méthodologie complète du test : HTML sémantique et compréhension par les LLM

Murielle Germain, redactionseo.com

Document de référence technique, associé à l'article : Le HTML sémantique est-il vraiment utile aux IA ? Mon test sur les tableaux de données (partie 1)

Juillet 2026

1. Contexte et objectif du test

Vérifier si le HTML sémantique (balises table, thead, th scope, caption, ainsi que les attributs id et headers pour les cas complexes) améliore la compréhension d'un tableau de données par des modèles de langage, par comparaison à un tableau visuellement identique construit uniquement avec des balises div.

Quatre sous-protocoles composent ce test :

- Test 1 : tableau simple, 5 lignes de données, 3 colonnes.
- Test 2 : tableau complexifié, 10 lignes réparties en 3 groupes, entouré de bruit de page (navigation, bannière cookies, publicité, articles connexes).
- Test 3 : vérification de contamination entre appels API, sur un contenu fictif distinct des tableaux SEO.
- Test 4 : tableau à double croisement (trimestre, site, métrique), avec attributs id/headers explicites contre absence totale d'association structurelle.

Un test de robustesse par répétition (5 exécutions du test d'arbitrage par modèle et par version) complète le protocole.

2. Modèles et paramètres API

Modèle	Identifiant exact	Méthode d'appel	Paramètres
Claude	claude-sonnet-4-6	API Anthropic, endpoint /v1/messages	max_tokens: 1000 (1200 pour tests 2 et 4), un seul tour, aucun historique
ChatGPT	gpt-4o	API OpenAI, endpoint /v1/chat/completions	max_tokens: 1000 (1200 pour tests 2 et 4), un seul tour
Gemini	gemini-2.5-flash	API Google, endpoint	un seul tour,

Modèle	Identifiant exact	Méthode d'appel	Paramètres
		generativelanguage.googleapis.com/v1beta	température par défaut de l'API
Gemma 27B	gemma-4-26B-A4Bit-UD-IQ4_XS.gguf	API locale, llama-server, endpoint compatible OpenAI (port 8085)	max_tokens: 1500, temperature: 0.2, repeat_penalty: 1.1

Configuration du serveur local (llama-server) :

```
/home/muriel/llama.cpp/build/bin/llama-server \
  -m /home/muriel/llama.cpp/models/gemma-4-26B-A4B-it-UD-IQ4_XS.gguf \
  --gpu-layers 99 -fa auto -c 8192 -ub 2048 -b 2048 -t 8 \
  --host 0.0.0.0 --port 8085 --reasoning-format none --jinja
```

Isolation entre appels : chaque combinaison modèle/version/question fait l'objet d'un appel API indépendant, sans historique de conversation ni contexte issu d'un appel précédent. Aucune mémoire n'est partagée entre deux appels, y compris entre la version A et la version B d'un même test. Cette isolation a été vérifiée explicitement par le Test 3 (voir section 6.3).

Délai entre appels : une pause de 8 secondes est appliquée entre chaque appel, pour respecter les limites de fréquence de l'API Gemini. En cas d'erreur HTTP 429 (trop de requêtes), une nouvelle tentative automatique est effectuée avec un délai croissant (20 s, 40 s, 60 s, jusqu'à 5 tentatives).

3. Les tableaux de test

3.1 Test 1 : tableau simple

Version A, HTML sémantique

```
<table>
  <caption>Comparaison des performances SEO de deux sites analysés en juin
2026</caption>
  <thead>
    <tr>
      <th scope="col">Indicateur</th>
      <th scope="col">Site A</th>
      <th scope="col">Site B</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">Pages indexées</th>
      <td>870</td>
      <td>1 250</td>
    </tr>
    <tr>
      <th scope="row">Trafic organique mensuel</th>
      <td>45 000 visites</td>
      <td>38 000 visites</td>
    </tr>
    <tr>
```

```
      <th scope="row">Score Core Web Vitals</th>
      <td>68/100</td>
      <td>92/100</td>
    </tr>
    <tr>
      <th scope="row">Taux de conversion</th>
      <td>3,2 %</td>
      <td>1,8 %</td>
    </tr>
    <tr>
      <th scope="row">Nombre de domaines référents</th>
      <td>340</td>
      <td>180</td>
    </tr>
  </tbody>
</table>
```

Version B, div

```
<div class="table">
  <div class="row">
    <div>Indicateur</div>
    <div>Site A</div>
    <div>Site B</div>
  </div>
  <div class="row">
    <div>Pages indexées</div>
    <div>870</div>
    <div>1 250</div>
  </div>
  <div class="row">
    <div>Trafic organique mensuel</div>
    <div>45 000 visites</div>
    <div>38 000 visites</div>
  </div>
  <div class="row">
    <div>Score Core Web Vitals</div>
    <div>68/100</div>
    <div>92/100</div>
  </div>
  <div class="row">
    <div>Taux de conversion</div>
    <div>3,2 %</div>
    <div>1,8 %</div>
  </div>
  <div class="row">
    <div>Nombre de domaines référents</div>
    <div>340</div>
    <div>180</div>
  </div>
</div>
```

3.2 Test 2 : tableau complexifié avec bruit de page

Version A, sémantique avec groupes, entourée de bruit de page

```
<header>
  <nav>
    <a href="/">Accueil</a> / <a href="/audits">Audits SEO</a> / Comparatif juin 2026
```

```
  </nav>
</header>
<div class="cookie-banner">Ce site utilise des cookies. <a href="#">En savoir
plus</a></div>
<main>
  <article>
    <h1>Comparatif détaillé, Site A vs Site B, juin 2026</h1>
    <p>Analyse complète des performances sur trois dimensions : acquisition,
    technique et conversion.</p>
    <table>
      <caption>Comparaison des performances SEO de deux sites analysés en juin
2026</caption>
      <thead>
        <tr>
          <th scope="col">Indicateur</th>
          <th scope="col">Site A</th>
          <th scope="col">Site B</th>
        </tr>
      </thead>
      <tbody>
        <tr><th colspan="3" scope="colgroup">Acquisition</th></tr>
        <tr>
          <th scope="row">Trafic organique mensuel</th>
          <td>45 000 visites</td>
          <td>38 000 visites</td>
        </tr>
        <tr>
          <th scope="row">Nombre de domaines référents</th>
          <td>340</td>
          <td>180</td>
        </tr>
      </tbody>
      <tbody>
        <tr><th colspan="3" scope="colgroup">Performance technique</th></tr>
        <tr>
          <th scope="row">Pages indexées</th>
          <td>870</td>
          <td>1 250</td>
        </tr>
        <tr>
          <th scope="row">Score Core Web Vitals</th>
          <td>68/100</td>
          <td>92/100</td>
        </tr>
        <tr>
          <th scope="row">Temps de chargement mobile</th>
          <td>3,8 s</td>
          <td>1,9 s</td>
        </tr>
      </tbody>
      <tbody>
        <tr><th colspan="3" scope="colgroup">Conversion</th></tr>
        <tr>
          <th scope="row">Taux de conversion</th>
          <td>3,2 %</td>
          <td>1,8 %</td>
        </tr>
        <tr>
          <th scope="row">Taux de rebond</th>
          <td>38 %</td>
          <td>52 %</td>
        </tr>
```

```
      </tbody>
    </table>
  </article>
  <aside>
    <div class="pub">Publicité : Formation SEO en 6 semaines, -20% ce mois-ci</div>
    <div class="articles-connexes">
      <h2>Articles connexes</h2>
      <ul>
        <li><a href="#">10 outils SEO gratuits en 2026</a></li>
        <li><a href="#">Comment auditer un site en 30 minutes</a></li>
        <li><a href="#">Étude de cas : +150 % de trafic en 8 mois</a></li>
      </ul>
    </div>
  </aside>
</main>
<footer>
  <p>Mentions légales | Contact | Plan du site | 2026</p>
</footer>
```

Version B, div avec groupes, même bruit de page

Le header, la bannière cookies, l'aside et le footer sont identiques à la version A. Seul le tableau change :

```
<div class="table">
  <div class="row header">
    <div>Indicateur</div>
    <div>Site A</div>
    <div>Site B</div>
  </div>
  <div class="row group-header">Acquisition</div>
  <div class="row">
    <div>Trafic organique mensuel</div>
    <div>45 000 visites</div>
    <div>38 000 visites</div>
  </div>
  <div class="row">
    <div>Nombre de domaines référents</div>
    <div>340</div>
    <div>180</div>
  </div>
  <div class="row group-header">Performance technique</div>
  <div class="row">
    <div>Pages indexées</div>
    <div>870</div>
    <div>1 250</div>
  </div>
  <div class="row">
    <div>Score Core Web Vitals</div>
    <div>68/100</div>
    <div>92/100</div>
  </div>
  <div class="row">
    <div>Temps de chargement mobile</div>
    <div>3,8 s</div>
    <div>1,9 s</div>
  </div>
  <div class="row group-header">Conversion</div>
  <div class="row">
    <div>Taux de conversion</div>
```

```
    <div>3,2 %</div>
    <div>1,8 %</div>
  </div>
  <div class="row">
    <div>Taux de rebond</div>
    <div>38 %</div>
    <div>52 %</div>
  </div>
</div>
```

3.3 Test 3 : vérification de contamination

Contenu volontairement distinct des tableaux SEO, pour qu'aucun modèle ne puisse connaître les valeurs par ailleurs.

Version A

```
<article>
  <h1>Rapport interne, projet Lievre-Bleu</h1>
  <p>Le code de reference du projet Lievre-Bleu pour le trimestre est 4471.</p>
</article>
```

Version B

```
<article>
  <h1>Rapport interne, projet Lievre-Bleu</h1>
  <p>Le code de reference du projet Lievre-Bleu pour le trimestre est 9902.</p>
</article>
```

3.4 Test 4 : tableau à double croisement (id/headers)

Version A, avec id et headers explicites

```
<table>
  <caption id="cap1">Trafic organique et taux de conversion par trimestre
  et par site, année 2026</caption>
  <thead>
    <tr>
      <th id="h-trimestre" rowspan="2" scope="col">Trimestre</th>
      <th id="h-sitea" colspan="2" scope="colgroup">Site A</th>
      <th id="h-siteb" colspan="2" scope="colgroup">Site B</th>
    </tr>
    <tr>
      <th id="h-sitea-trafic" headers="h-sitea">Trafic organique</th>
      <th id="h-sitea-conv" headers="h-sitea">Taux de conversion</th>
      <th id="h-siteb-trafic" headers="h-siteb">Trafic organique</th>
      <th id="h-siteb-conv" headers="h-siteb">Taux de conversion</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th id="h-t1" headers="h-trimestre" scope="row">T1 2026</th>
      <td headers="h-t1 h-sitea h-sitea-trafic">32 000</td>
      <td headers="h-t1 h-sitea h-sitea-conv">2,1 %</td>
      <td headers="h-t1 h-siteb h-siteb-trafic">28 000</td>
      <td headers="h-t1 h-siteb h-siteb-conv">3,4 %</td>
    </tr>
```

```
    <tr>
      <th id="h-t2" headers="h-trimestre" scope="row">T2 2026</th>
      <td headers="h-t2 h-sitea h-sitea-trafic">35 000</td>
      <td headers="h-t2 h-sitea h-sitea-conv">2,4 %</td>
      <td headers="h-t2 h-siteb h-siteb-trafic">30 000</td>
      <td headers="h-t2 h-siteb h-siteb-conv">3,1 %</td>
    </tr>
    <tr>
      <th id="h-t3" headers="h-trimestre" scope="row">T3 2026</th>
      <td headers="h-t3 h-sitea h-sitea-trafic">41 000</td>
      <td headers="h-t3 h-sitea h-sitea-conv">2,9 %</td>
      <td headers="h-t3 h-siteb h-siteb-trafic">33 000</td>
      <td headers="h-t3 h-siteb h-siteb-conv">2,8 %</td>
    </tr>
    <tr>
      <th id="h-t4" headers="h-trimestre" scope="row">T4 2026</th>
      <td headers="h-t4 h-sitea h-sitea-trafic">45 000</td>
      <td headers="h-t4 h-sitea h-sitea-conv">3,2 %</td>
      <td headers="h-t4 h-siteb h-siteb-trafic">38 000</td>
      <td headers="h-t4 h-siteb h-siteb-conv">1,8 %</td>
    </tr>
  </tbody>
</table>
```

Version B, sans association explicite (div)

```
<div class="tableau">
  <div class="titre">Trafic organique et taux de conversion par trimestre
  et par site, année 2026</div>
  <div class="ligne entete">
    <div>Trimestre</div>
    <div class="groupe">Site A</div>
    <div class="groupe">Site B</div>
  </div>
  <div class="ligne sous-entete">
    <div></div>
    <div>Trafic organique</div>
    <div>Taux de conversion</div>
    <div>Trafic organique</div>
    <div>Taux de conversion</div>
  </div>
  <div class="ligne">
    <div>T1 2026</div><div>32 000</div><div>2,1 %</div><div>28 000</div><div>3,4
%</div>
  </div>
  <div class="ligne">
    <div>T2 2026</div><div>35 000</div><div>2,4 %</div><div>30 000</div><div>3,1
%</div>
  </div>
  <div class="ligne">
    <div>T3 2026</div><div>41 000</div><div>2,9 %</div><div>33 000</div><div>2,8
%</div>
  </div>
  <div class="ligne">
    <div>T4 2026</div><div>45 000</div><div>3,2 %</div><div>38 000</div><div>1,8
%</div>
  </div>
</div>
```

4. Les prompts exacts

Chaque prompt est construit selon le même gabarit : le texte « Voici le contenu d'une page web au format HTML. », suivi du contenu HTML complet, suivi de la question. Seule la question varie.

```
Voici le contenu d'une page web au format HTML.

{html_content}

{question}
```

4.1 Test 1 (tableau simple)

- Extraction : Extrais toutes les données de ce tableau sous forme de tableau Markdown.
- Raisonnement avec arbitrage : Quel site présente les meilleures performances SEO ? Justifie ta réponse avec les données du tableau.
- Question ciblée : Quel est le taux de conversion du Site A ?
- Résumé : Résume les différences principales entre les deux sites.

4.2 Test 2 (tableau complexe et bruit)

- Extraction : Extrais toutes les données du tableau de comparaison sous forme de tableau Markdown, en conservant les trois groupes (Acquisition, Performance technique, Conversion).
- Raisonnement : Quel site présente la meilleure performance SEO globale ? Justifie ta réponse en tenant compte des trois groupes de critères.
- Question ciblée : Quel est le temps de chargement mobile du Site B ?
- Résumé filtrant le bruit : Résume les différences principales entre les deux sites en te basant uniquement sur le tableau de comparaison, sans tenir compte des articles connexes ou de la publicité.

4.3 Test 3 (contamination)

- Question unique : Quel est le code de référence du projet Lievre-Bleu mentionné dans cette page ?

4.4 Test 4 (double croisement id/headers)

- Extraction : Extrais toutes les données de ce tableau sous forme de tableau Markdown.
- Question ciblée 1 : Quel est le taux de conversion du Site B au T3 2026 ?
- Question ciblée 2 : Quel est le trafic organique du Site A au T4 2026 ?
- Résumé de série : Résume l'évolution du taux de conversion du Site B sur les quatre trimestres de 2026.

4.5 Répétition (robustesse, 5 exécutions)

Reprend la question de raisonnement du Test 2, avec une consigne de format ajoutée :

```
Quel site presente la meilleure performance SEO globale ? Justifie ta reponse
en tenant compte des trois groupes de criteres. Termine ta reponse par une ligne,
```

```
seule sur sa ligne, exactement au format : VERDICT_FINAL: Site A ou VERDICT_FINAL: Site
B
```

5. Le script de test

5.1 Script principal (Test 1, structure de référence)

```
import os
import json
import time
import urllib.request
import urllib.error
from datetime import datetime
from pathlib import Path

ANTHROPIC_API_KEY = os.environ.get("ANTHROPIC_API_KEY")
OPENAI_API_KEY = os.environ.get("OPENAI_API_KEY")
GOOGLE_API_KEY = os.environ.get("GOOGLE_API_KEY")
LLAMA_SERVER_URL = os.environ.get("LLAMA_SERVER_URL",
"http://localhost:8085/completion")

ANTHROPIC_MODEL = "claude-sonnet-4-6"
OPENAI_MODEL = "gpt-4o"
GOOGLE_MODEL = "gemini-2.5-flash"

RESULTS_DIR = Path("resultats")
RESULTS_DIR.mkdir(exist_ok=True)

PROMPTS = {
    "test1_extraction": "Extrais toutes les donnees de ce tableau sous forme de tableau
Markdown.",
    "test2_raisonnement": "Quel site presente les meilleures performances SEO ?
Justifie ta reponse avec les donnees du tableau.",
    "test3_cible": "Quel est le taux de conversion du Site A ?",
    "test4_resume": "Resume les differences principales entre les deux sites.",
}

def load_table(version):
    path = Path(f"table_{version}.html")
    return path.read_text(encoding="utf-8")

def build_prompt(html_content, question):
    return f"Voici le contenu d'une page web au format HTML.\n\n{html_content}\n\
n{question}"

def call_with_retry(fn, max_retries=5, base_wait=20):
    last_error = None
    for attempt in range(max_retries):
        try:
            return fn()
        except urllib.error.HTTPError as e:
            if e.code == 429:
                wait = base_wait * (attempt + 1)
                time.sleep(wait)
                last_error = "HTTP Error 429: Too Many Requests"
            else:
                return f"ERREUR : HTTP Error {e.code}: {e.reason}"
    return f"ERREUR apres {max_retries} tentatives : {last_error}"
```

```
def call_anthropic(prompt):
    if not ANTHROPIC_API_KEY:
        return "ERREUR : ANTHROPIC_API_KEY absente."
    def do_call():
        req = urllib.request.Request(
            "https://api.anthropic.com/v1/messages",
            data=json.dumps({
                "model": ANTHROPIC_MODEL,
                "max_tokens": 1000,
                "messages": [{"role": "user", "content": prompt}],
            }).encode("utf-8"),
            headers={
                "content-type": "application/json",
                "x-api-key": ANTHROPIC_API_KEY,
                "anthropic-version": "2023-06-01",
            },
            method="POST",
        )
        with urllib.request.urlopen(req) as resp:
            data = json.loads(resp.read())
        return "".join(block.get("text", "") for block in data.get("content", []))
    return call_with_retry(do_call)

def call_openai(prompt):
    if not OPENAI_API_KEY:
        return "ERREUR : OPENAI_API_KEY absente."
    def do_call():
        req = urllib.request.Request(
            "https://api.openai.com/v1/chat/completions",
            data=json.dumps({
                "model": OPENAI_MODEL,
                "max_tokens": 1000,
                "messages": [{"role": "user", "content": prompt}],
            }).encode("utf-8"),
            headers={
                "content-type": "application/json",
                "authorization": f"Bearer {OPENAI_API_KEY}",
            },
            method="POST",
        )
        with urllib.request.urlopen(req) as resp:
            data = json.loads(resp.read())
        return data["choices"][0]["message"]["content"]
    return call_with_retry(do_call)

def call_google(prompt):
    if not GOOGLE_API_KEY:
        return "ERREUR : GOOGLE_API_KEY absente."
    def do_call():
        url =
f"https://generativelanguage.googleapis.com/v1beta/models/{GOOGLE_MODEL}:generateConten
t?key={GOOGLE_API_KEY}"
        req = urllib.request.Request(
            url,
            data=json.dumps({"contents": [{"parts": [{"text": prompt}]}]}).encode("utf-
8"),
            headers={"content-type": "application/json"},
            method="POST",
        )
        with urllib.request.urlopen(req) as resp:
            data = json.loads(resp.read())
```

```
        return data["candidates"][0]["content"]["parts"][0]["text"]
    return call_with_retry(do_call, base_wait=30)

def call_llama_local(prompt):
    chat_url = LLAMA_SERVER_URL.replace("/completion", "/v1/chat/completions")
    req = urllib.request.Request(
        chat_url,
        data=json.dumps({
            "messages": [{"role": "user", "content": prompt}],
            "max_tokens": 1500,
            "temperature": 0.2,
            "repeat_penalty": 1.1,
        }).encode("utf-8"),
        headers={"content-type": "application/json"},
        method="POST",
    )
    try:
        with urllib.request.urlopen(req) as resp:
            data = json.loads(resp.read())
        raw_content = data["choices"][0]["message"]["content"]
        if "<channel|>" in raw_content:
            return raw_content.split("<channel|>", 1)[1].strip()
        return raw_content.strip()
    except Exception as e:
        return f"ERREUR llama-server : {e}"

MODELS = {
    "claude": call_anthropic,
    "chatgpt": call_openai,
    "gemini": call_google,
    "gemma_local": call_llama_local,
}

def run():
    timestamp = datetime.now().strftime("%Y%m%d-%H%M%S")
    for version in ["a", "b"]:
        html_content = load_table(version)
        for test_name, question in PROMPTS.items():
            prompt = build_prompt(html_content, question)
            for model_name, model_fn in MODELS.items():
                try:
                    response = model_fn(prompt)
                except Exception as e:
                    response = f"ERREUR : {e}"
                out_path = RESULTS_DIR /
f"{model_name}_v{version}_{test_name}_{timestamp}.txt"
                out_path.write_text(
                    f"Modele : {model_name}\nVersion : {version.upper()}\nTest :
{test_name}\n"
                    f"Date : {timestamp}\nPrompt envoye :\n{prompt}\n\nReponse :\
n{response}",
                    encoding="utf-8",
                )
                time.sleep(8)

if __name__ == "__main__":
    run()
```

Note : les scripts des Tests 2, 3 et 4 reprennent la même architecture (mêmes fonctions d'appel API, même logique de nouvelle tentative), avec un dictionnaire PROMPTS et un chargement de fichier HTML adaptés à chaque test.

5.2 Script du Test 3 (contamination), spécificités

```
CONTENU_A = """
<article>
  <h1>Rapport interne, projet Lievre-Bleu</h1>
  <p>Le code de reference du projet Lievre-Bleu pour le trimestre est 4471.</p>
</article>
"""

CONTENU_B = """
<article>
  <h1>Rapport interne, projet Lievre-Bleu</h1>
  <p>Le code de reference du projet Lievre-Bleu pour le trimestre est 9902.</p>
</article>
"""

QUESTION = "Quel est le code de reference du projet Lievre-Bleu mentionne dans cette
page ?"

def run():
    resultats = {}
    for version, contenu, attendu in [("A", CONTENU_A, "4471"), ("B", CONTENU_B,
"9902")]:
        prompt = build_prompt(contenu, QUESTION)
        for model_name, model_fn in MODELS.items():
            reponse = model_fn(prompt)
            resultats[f"{model_name}_{version}"] = reponse
            time.sleep(8)
    for model_name in MODELS:
        rep_a = resultats.get(f"{model_name}_A", "")
        rep_b = resultats.get(f"{model_name}_B", "")
        a_ok = "4471" in rep_a
        b_ok = "9902" in rep_b
        a_contamine = "9902" in rep_a
        b_contamine = "4471" in rep_b
        print(f"{model_name} : A correct={a_ok}, contamination par B={a_contamine} | "
              f"B correct={b_ok}, contamination par A={b_contamine}")
```

5.3 Script de répétition (robustesse, 5 exécutions), spécificités

```
REPETITIONS = 5

def extraire_verdict(reponse):
    import re
    match = re.search(r"VERDICT_FINAL\s*:\s*Site\s*([AB])", reponse, re.IGNORECASE)
    return f"Site {match.group(1).upper()}" if match else "NON_DETECTE"

def run():
    verdicts = {model: {"a": [], "b": []} for model in MODELS}
    for version in ["a", "b"]:
        html_content = load_page(version)
        prompt = build_prompt(html_content)
        for model_name, model_fn in MODELS.items():
            for rep in range(1, REPETITIONS + 1):
                response = model_fn(prompt)
                verdict = extraire_verdict(response)
```

```
            verdicts[model_name][version].append(verdict)
            time.sleep(8)
```

6. Les sorties brutes

6.1 Test 1 : tableau simple

Note de transparence : l'ensemble des réponses de ce test a été récupéré et vérifié mot pour mot à partir du fichier source. Une anomalie technique initiale (erreur HTTP 503 sur l'appel Gemini / version B / extraction) a été identifiée puis résolue par une nouvelle tentative isolée, documentée ci-dessous.

Extraction brute

Sur les 8 combinaisons possibles (4 modèles, 2 versions), les 8 extractions sont exactes et conformes au tableau source, sans erreur de chiffre ni omission.

Claude, versions A et B : extraction identique et exacte des 5 lignes. ChatGPT, versions A et B : extraction exacte (différence typographique mineure en version B : « 32,000 » au lieu de « 32 000 » pour le séparateur de milliers, sans incidence sur l'exactitude). Gemini, version A : extraction exacte. Gemini, version B : première tentative en erreur (HTTP 503, service indisponible). Une nouvelle tentative isolée, effectuée séparément après résolution du service, a produit une extraction exacte et complète des 5 lignes, identique à la version A. Gemma local, versions A et B : extraction exacte des 5 lignes.

Raisonnement avec arbitrage

Les 4 modèles concluent, sur les deux versions, que le Site A présente globalement les meilleures performances SEO, en reconnaissant explicitement la supériorité technique du Site B (Core Web Vitals, pages indexées) sans que cela renverse le verdict global. Aucune divergence de verdict entre la version A et la version B, pour aucun des 4 modèles.

Exemple représentatif (Claude, version A) :

« Le Site A présente globalement les meilleures performances SEO, car il génère davantage de trafic organique, dispose d'un profil de liens bien plus solide et affiche un taux de conversion nettement supérieur. Cependant, son score Core Web Vitals (68/100) est insuffisant et constitue un point de vigilance, car Google en tient compte dans son classement. »

Exemple représentatif (Gemma local, version A), qui formule sa réponse comme un arbitrage conditionnel plutôt qu'un verdict unique, mais aboutit à la même hiérarchie factuelle :

« Il n'y a pas de réponse unique, car le choix du “meilleur” site dépend des objectifs stratégiques [...] Choisissez le Site A si vous privilégiez la rentabilité immédiate, l'autorité et le volume de trafic. Choisissez le Site B si vous privilégiez la fondation technique et le potentiel de croissance. »

Question ciblée (taux de conversion du Site A)

Réponse strictement correcte (3,2 %) sur les 4 modèles et les 2 versions, sans exception :

Modèle	Version A	Version B
Claude	3,2 % (correct)	3,2 % (correct)
ChatGPT	3,2 % (correct)	3,2 % (correct)
Gemini	3,2 % (correct)	3,2 % (correct)
Gemma local	3,2 % (correct)	3,2 % (correct)

Résumé

Les 4 modèles produisent, sur les deux versions, un résumé identifiant les mêmes contrastes factuels sans erreur : Site A meilleur en trafic organique, taux de conversion et domaines référents ; Site B meilleur en pages indexées et score Core Web Vitals. Claude et Gemma local ajoutent une synthèse tabulaire ou thématique ; ChatGPT et Gemini restent en prose structurée par points numérotés. Aucune inversion de valeur, aucune confusion entre sites, sur les 8 réponses.

6.2 Test 2 : tableau complexe avec bruit de page

Extraction avec groupes (4 modèles, 2 versions)

Claude, ChatGPT, Gemini et Gemma local restituent chacun l'intégralité des 3 groupes (Acquisition, Performance technique, Conversion) et des 7 indicateurs, sur les deux versions, sans perte de donnée ni confusion de groupe. Aucune fuite du bruit de page (publicité, articles connexes) n'a été observée dans les extractions.

Raisonnement (arbitrage sur 3 groupes)

Les 4 modèles identifient correctement que le Site A domine les groupes Acquisition et Conversion, tandis que le Site B domine le groupe Performance technique, sur les deux versions. Le verdict global diverge selon le modèle (Claude et Gemma concluent Site A globalement meilleur en pondérant acquisition/conversion comme prioritaires ; le détail complet est disponible dans les fichiers sources), mais chaque modèle reste cohérent avec lui-même entre la version A et la version B.

Question ciblée (temps de chargement mobile du Site B)

Réponse strictement identique et correcte (1,9 s ou 1,9 seconde selon la formulation du modèle) sur les 4 modèles et les 2 versions :

Modèle	Version A	Version B
Claude	« 1,9 seconde »	« 1,9 seconde »
ChatGPT	« 1,9 secondes »	« 1,9 secondes »
Gemini	« 1,9 s »	« 1,9 s »
Gemma local	« 1,9 s »	« 1,9 s »

Résumé en filtrant le bruit

Les 4 modèles produisent un résumé complet des trois groupes, sans mentionner la publicité ni les articles connexes présents dans la page, sur les deux versions. Exemple représentatif (Gemma local, version B) :

« Voici un résumé des principales différences entre le Site A et le Site B, classées par thématique : 1. Performance SEO et Autorité (Avantage Site A) [...] 2. Performance Technique et Expérience Utilisateur (Avantage Site B) [...] 3. Efficacité Commerciale (Avantage Site A) [...] 4. Volume de Contenu (Avantage Site B) [...] »

6.3 Test 3 : vérification de contamination

Réponses obtenues (extrait représentatif, 4 modèles). Attendu : version A donne 4471, version B donne 9902.

```
Appel : claude / version A (attendu : 4471)
    Reponse : D'après le contenu de la page web, le code de référence du
    projet Lièvre-Bleu pour le trimestre est 4471.

Appel : chatgpt / version A (attendu : 4471)
    Reponse : Le code de référence du projet Lievre-Bleu mentionné dans
    cette page est 4471.

Appel : gemini / version A (attendu : 4471)
    Reponse : Le code de référence du projet Lievre-Bleu mentionné dans
    cette page est 4471.

Appel : gemma_local / version A (attendu : 4471)
    Reponse : Le code de référence du projet Lievre-Bleu mentionné dans
    cette page est 4471.

Appel : claude / version B (attendu : 9902)
    Reponse : D'après le contenu de la page web, le code de référence du
    projet Lièvre-Bleu pour le trimestre est 9902.

Appel : chatgpt / version B (attendu : 9902)
    Reponse : Le code de référence du projet Lievre-Bleu mentionné dans
    cette page est 9902.

Appel : gemini / version B (attendu : 9902)
    Reponse : Le code de référence du projet Lievre-Bleu mentionné dans
    cette page est 9902.

Appel : gemma_local / version B (attendu : 9902)
    Reponse : Le code de référence du projet Lievre-Bleu mentionné dans
    cette page est 9902.
```

Récapitulatif : sur les 4 modèles et les 2 versions, chaque réponse correspond exactement à la valeur de sa propre version. Aucune contamination détectée (aucun modèle ne mentionne 9902 en version A, ni 4471 en version B).

6.4 Test 4 : tableau à double croisement (id/headers)

Extraction complète (4 modèles, 2 versions)

Les 4 modèles restituent les 4 trimestres avec les 4 valeurs associées (trafic et conversion pour Site A et Site B), sans erreur, sur les deux versions.

Question ciblée 1 (taux de conversion du Site B au T3 2026, attendu 2,8 %)

Modèle	Version A	Version B
Claude	2,8 % (correct)	2,8 % (correct)
ChatGPT	2,8 % (correct)	2,8 % (correct)
Gemini	2,8 % (correct, avec détail du raisonnement de croisement ligne/colonne)	2,8 % (correct, réponse directe sans raisonnement affiché)
Gemma local	2,8 % (correct)	2,8 % (correct)

Réponse détaillée de Gemini, version A (seul cas où un modèle détaille son raisonnement de croisement) :

« Pour trouver le taux de conversion du Site B au T3 2026, nous devons localiser la ligne correspondant au “T3 2026” et la colonne correspondant au “Taux de conversion” sous “Site B”. [...] La colonne du “Taux de conversion” pour “Site B” est identifiée par les en-têtes h-siteb et h-siteb-conv. En croisant la ligne “T3 2026” et la colonne “Taux de conversion” pour “Site B”, nous trouvons la valeur : 2,8 %. »

Question ciblée 2 (trafic organique du Site A au T4 2026, attendu 45 000)

Réponse correcte (« 45 000 ») sur les 4 modèles et les 2 versions, sans exception.

Résumé d'une série temporelle (évolution du taux de conversion du Site B)

Les 4 modèles décrivent correctement la tendance baissière (3,4 % puis 3,1 % puis 2,8 % puis 1,8 %) sur les 4 trimestres, sur les deux versions, avec une variation dans le niveau de détail (certains modèles chiffrent l'écart point par point, d'autres se limitent à la tendance générale) mais aucune erreur de valeur.

6.5 Répétitions (robustesse, 5 exécutions)

Récapitulatif des verdicts sur le test d'arbitrage, répété 5 fois par modèle et par version :

Modèle	Version A	Version B
Claude	Site A : 5 fois sur 5	Site A : 5 fois sur 5
ChatGPT	Site A : 5 fois sur 5	Site A : 5 fois sur 5
Gemini	Site A : 4 fois, Site B : 1 fois	Site A : 4 fois, Site B : 1 fois
Gemma local	Site A : 2 fois, non détecté : 3 fois	Site A : 2 fois, non détecté : 3 fois

Lecture : pour chaque modèle, la répartition des verdicts est identique entre la version A et la version B. La variabilité observée chez Gemini (4 contre 1) et chez Gemma local (2 verdicts détectés sur 5, le modèle ne respectant pas toujours le format de sortie demandé) est répartie de façon égale entre les deux versions, ce qui indique un aléa de génération propre à chaque modèle plutôt qu'un effet de la structure HTML testée.

7. Synthèse des résultats

Test	Modèle	Version	Question	Résultat	Remarque
1	Claude	A / B	Extraction	Correct / Correct	

Test	Modèle	Version	Question	Résultat	Remarque
1	ChatGPT	A / B	Extraction	Correct / Correct	Formatage des milliers différent en B (32,000 vs 32 000), sans incidence
1	Gemini	A	Extraction	Correct	
1	Gemini	B	Extraction	Correct (après nouvelle tentative)	Première tentative en erreur 503, résolue par un appel isolé postérieur
1	Gemma local	A / B	Extraction	Correct / Correct	
1	4 modèles	A / B	Arbitrage	Cohérent / Cohérent	Verdict « Site A » chez les 4 modèles, sur les 2 versions
1	4 modèles	A / B	Question ciblée	Correct / Correct	3,2 %, identique sur les 8 réponses
1	4 modèles	A / B	Résumé	Correct / Correct	Contrastes identiques sur les 8 réponses
2	4 modèles	A / B	Extraction avec groupes	Correct / Correct	Aucune perte de groupe ni de valeur
2	4 modèles	A / B	Arbitrage 3 groupes	Cohérent / Cohérent	Verdict stable par modèle entre versions
2	4 modèles	A / B	Question ciblée	Correct / Correct	1,9 s, identique sur les 8 réponses
2	4 modèles	A / B	Résumé filtrant le bruit	Correct / Correct	Aucune fuite de bruit de page
3	4 modèles	A / B	Code de référence	Correct / Correct	Aucune contamination détectée
4	4 modèles	A / B	Extraction	Correct / Correct	
4	4 modèles	A / B	Question ciblée 1 (2,8 %)	Correct / Correct	Gemini détaille son raisonnement en A uniquement, réponse juste dans les deux cas
4	4 modèles	A / B	Question ciblée 2 (45 000)	Correct / Correct	
4	4 modèles	A / B	Résumé de série	Correct / Correct	
Répétition	Claude / ChatGPT	A / B	Arbitrage ×5	Stable (5/5)	Verdict identique dans les deux versions
Répétition	Gemini	A / B	Arbitrage ×5	Stable (4/5)	Répartition

Test	Modèle	Version	Question	Résultat	Remarque
					identique entre versions
Répétition	Gemma local	A / B	Arbitrage ×5	Partiellement non détecté (2/5)	Format de sortie non respecté 3 fois sur 5, également réparti entre versions

Conclusion consolidée

Sur l'ensemble des 37 cellules du tableau ci-dessus, toutes sont vérifiées avec certitude sur la base des fichiers de résultats bruts, et indiquent une absence de différence de justesse attribuable à la structure HTML (sémantique contre div) sur les quatre modèles testés. Une anomalie technique a été rencontrée en cours de protocole (erreur 503 de Gemini, extraction, Test 1, version B), sans lien avec la structure du contenu testé ; elle a été résolue par une nouvelle tentative isolée, dont le résultat (extraction exacte et complète) est cohérent avec l'ensemble des autres observations.

Document généré à partir des fichiers de résultats bruts du protocole. Pour toute question sur la reproductibilité de ce test, le code source complet est fourni en section 5.