



SPÉ NSI · TERMINALE

✓ EXIGIBLE AU BAC

PRIORITÉ P1

Langage SQL

requêtes de sélection, jointures, agrégations

🎯 À LA FIN DE CETTE FICHE, TU SAURAS...

- ✓ Écrire une requête SQL de sélection avec les clauses SELECT, FROM et WHERE.
- ✓ Écrire une requête SQL réalisant une jointure entre deux tables.
- ✓ Utiliser une fonction d'agrégation avec la clause GROUP BY pour calculer une valeur par groupe.

TON PARCOURS DE RÉVISION — coche chaque partie une fois révisée !

- | | |
|--|---|
| ☐ 1 · Les mots à connaître | ☐ 7 · La carte mentale |
| ☐ 2 · Les règles essentielles | ☐ 8 · Trouve l'erreur |
| ☐ 3 · Méthode | ☐ 9 · Exercices type Bac |
| ☐ 4 · Exemple type Bac | ☐ 10 · QCM chrono (5 min) |
| ☐ 5 · Les pièges à éviter | ☐ 11 · Ta carte « 30 secondes » |
| ☐ 6 · Tips pour t'en souvenir | ☐ ∞ Corrections (pour les parents) |

👤 LE COIN DES PARENTS — pas besoin de connaître le cours, tout est justifié.

- Aider à répartir les révisions de ce chapitre sur plusieurs jours (planning de répétition espacée J+1/J+3/J+7/J+14), sans se prononcer sur le contenu.
- Noter ensemble les dates clés du trimestre (DS, épreuves de spécialité) sur un support partagé (agenda, application).
- Vérifier que l'élève dispose d'un espace et d'un moment calmes pour réviser, et ajuster l'environnement si besoin (bruit, écrans, horaires).

1 Les mots à connaître *surligne ceux que tu confonds !*

SQL — le langage utilisé pour interroger et manipuler des données organisées en tables reliées entre elles ; une requête SQL est une instruction écrite dans ce langage qui décrit les données à extraire d'une ou plusieurs tables.

Clé primaire — un attribut, ou un groupe d'attributs, dont la valeur identifie de façon unique chaque n-uplet d'une table.

Clé étrangère — un attribut d'une table dont la valeur reprend la clé primaire d'une autre table, pour établir un lien entre les deux tables.

Clause SELECT — la partie d'une requête SQL qui indique quels attributs afficher dans le résultat.

Clause FROM — la partie d'une requête SQL qui indique la ou les tables interrogées.

Clause WHERE — la partie d'une requête SQL qui filtre les n-uplets selon une condition, en ne gardant que ceux qui la vérifient.

Jointure SQL — la combinaison de deux tables dans une requête SQL, réalisée en indiquant dans la clause WHERE l'égalité entre une clé primaire et la clé étrangère qui la référence.

Fonction d'agrégation — une fonction SQL qui calcule une valeur unique à partir d'un ensemble de n-uplets, par exemple compter (COUNT), additionner (SUM) ou faire une moyenne (AVG).

Clause GROUP BY — la partie d'une requête SQL qui regroupe les n-uplets partageant une même valeur d'attribut, avant d'appliquer une fonction d'agrégation à chaque groupe.

Table — un ensemble organisé en lignes et en colonnes qui stocke les données d'un même type d'objet.

Attribut — une colonne d'une table, qui décrit une caractéristique des objets stockés.

N-uplet — une ligne d'une table, qui représente un objet précis avec une valeur pour chacun de ses attributs.

2 Les règles essentielles *chaque règle avec son exemple*

- 1 **Une requête SQL de sélection simple s'écrit avec la clause SELECT (attributs à afficher), la clause FROM (table interrogée) et la clause WHERE (condition de filtre).**

exemple → `SELECT nom FROM Eleve WHERE classe = 'Terminale NSI'` affiche le nom des élèves de cette classe.

- 2 **La clause SELECT indique les attributs à afficher dans le résultat de la requête ; la clause FROM indique la table (ou les tables) interrogées.**

exemple → Dans `SELECT nom, classe FROM Eleve`, la clause SELECT liste nom et classe, la clause FROM désigne la table Eleve.

- 3 **La clause WHERE filtre les n-uplets selon une condition, en ne gardant que ceux qui la vérifient.**

exemple → `WHERE valeur > 15` ne garde que les n-uplets où l'attribut valeur dépasse 15.

- 4 **Une jointure SQL combine deux tables en indiquant, dans la clause WHERE, l'égalité entre une clé primaire et la clé étrangère qui la référence.**

exemple → `SELECT nom FROM Eleve, Note WHERE Eleve.identifiant = Note.identifiant_eleve AND Note.valeur > 15` combine les deux tables sur leur attribut commun.

- 5 **Une fonction d'agrégation calcule une valeur unique à partir d'un ensemble de n-uplets ; combinée à la clause GROUP BY, elle calcule cette valeur séparément pour chaque groupe de n-uplets partageant une même valeur d'attribut.**

exemple → `SELECT classe, COUNT(*) FROM Eleve GROUP BY classe` donne le nombre d'élèves pour chaque classe.

3

Méthode — Construire une requête SQL avec sélection, jointure et agrégation

Exemple suivi : donner le nombre d'élèves par classe, table Eleve (identifiant, nom, classe).

Étape 1 Écrire la clause SELECT avec les attributs à afficher. Ici, classe et une fonction d'agrégation COUNT(*).

Étape 2 Écrire la clause FROM avec la ou les tables interrogées. Ici, la table Eleve.

Étape 3 Écrire la clause WHERE avec la condition de filtre, en incluant l'égalité entre clé primaire et clé étrangère si une jointure est nécessaire. Ici, aucune condition WHERE n'est nécessaire.

Étape 4 Ajouter une fonction d'agrégation et une clause GROUP BY si un calcul par groupe est demandé. Ici, GROUP BY classe.

Étape 5 Vérifier que chaque attribut affiché en SELECT est soit un attribut du GROUP BY, soit le résultat d'une fonction d'agrégation : classe est dans le GROUP BY, COUNT(*) est une agrégation.

4

Exemple type Bac — rédigé pas à pas

Écrire la requête SQL qui donne le nombre d'élèves par classe, à partir de la table Eleve (identifiant, nom, classe).

Étape 1 — Écrire la clause SELECT avec l'attribut classe et la fonction d'agrégation COUNT(*).

Étape 2 — Écrire la clause FROM avec la table Eleve.

Étape 3 — Ajouter la clause GROUP BY sur l'attribut classe, pour calculer le compte séparément par classe.

Conclusion : La requête **SELECT classe, COUNT(*) FROM Eleve GROUP BY classe** donne le nombre d'élèves pour chaque classe.

5

Les pièges à éviter *les erreurs classiques en DS...*

✗ Oublier la clause WHERE nécessaire à une jointure SQL classique (égalité entre clé primaire et clé étrangère), ce qui multiplie les combinaisons de lignes sans lien réel entre elles.

✗ Afficher, dans la clause SELECT, un attribut qui n'est ni dans la clause GROUP BY ni le résultat d'une fonction d'agrégation.

✗ Confondre la clause WHERE (filtre avant tout regroupement) avec la clause GROUP BY (qui regroupe les n-uplets par valeur d'attribut).

6 Tips — pour t'en souvenir

SELECT = quoi afficher ; FROM = où chercher ; WHERE = quelle condition : retiens l'ordre logique de lecture.

Jointure SQL = une égalité clé primaire / clé étrangère dans la clause WHERE, jamais une coïncidence.

GROUP BY : chaque attribut du SELECT doit être dans le GROUP BY ou agrégé (COUNT, SUM, AVG), jamais les deux à la fois manquants.

7 La carte mentale *dessiner, c'est mémoriser !*

« LANGAGE SQL » au centre, 3 branches colorées : SELECT/FROM/WHERE (attributs, table, condition), JOINTURE SQL (égalité clé primaire / clé étrangère), AGRÉGATION (COUNT/SUM/AVG avec GROUP BY). Un exemple par branche. Redessine-la de mémoire après avoir lu la fiche.



ta carte mentale, de mémoire...

8 À toi de jouer — trouve l'erreur

Un élève écrit : « `SELECT nom, classe, COUNT(*) FROM Eleve GROUP BY classe` » pour compter les élèves par classe, en affichant aussi leur nom.

→ *cherche l'erreur avant de lire la réponse !*

L'erreur : L'attribut nom n'est ni dans la clause GROUP BY (qui ne porte que sur classe) ni le résultat d'une fonction d'agrégation : pour chaque classe regroupée, il existe plusieurs noms d'élèves possibles, la requête ne peut donc pas afficher un seul nom par groupe de façon cohérente. Chaque attribut de la clause SELECT doit soit figurer dans la clause GROUP BY, soit être le résultat d'une fonction d'agrégation (COUNT, SUM, AVG...) : sinon, la requête ne sait pas quelle valeur choisir parmi les plusieurs n-uplets d'un même groupe.

Exercice 1. Écrire la requête SQL qui affiche le nom des élèves de la classe 'Terminale NSI', à partir de la table Eleve (identifiant, nom, classe).

Exercice 2. Écrire la requête SQL qui donne la moyenne des notes par élève, à partir de la table Note (identifiant_note, identifiant_eleve, valeur).

Une seule bonne réponse par question, sans calculatrice — entraînement épreuve d'automatismes du Bac 2026.

1. La clause SELECT d'une requête SQL indique :

- ☐ A les attributs à afficher dans le résultat
- ☐ B la table interrogée
- ☐ C la condition de filtre des n-uplets

2. La clause FROM d'une requête SQL indique :

- ☐ A les attributs à afficher
- ☐ B la ou les tables interrogées
- ☐ C le regroupement des n-uplets

3. La clause WHERE d'une requête SQL sert à :

- ☐ A afficher certains attributs
- ☐ B regrouper les n-uplets par valeur d'attribut
- ☐ C filtrer les n-uplets selon une condition

4. Réaliser une jointure SQL entre deux tables impose typiquement d'écrire, dans la clause WHERE :

- ☐ A l'égalité entre une clé primaire et la clé étrangère qui la référence
- ☐ B une comparaison entre deux attributs de la même table
- ☐ C aucune condition, la jointure se fait automatiquement

5. Une fonction d'agrégation calcule :

- ☐ A une nouvelle table entière, ligne par ligne
- ☐ B une valeur unique à partir d'un ensemble de n-uplets
- ☐ C le regroupement des n-uplets selon une valeur commune

6. La clause GROUP BY sert à :

- ☐ A filtrer les n-uplets avant tout regroupement
- ☐ B afficher un seul attribut, sans condition
- ☐ C regrouper les n-uplets partageant une même valeur d'attribut, avant d'agréger chaque groupe

7. Dans SELECT classe, COUNT(*) FROM Eleve GROUP BY classe, le résultat donne :

- ☐ A le nombre d'élèves pour chaque classe
- ☐ B le nombre total d'élèves de l'établissement, toutes classes confondues
- ☐ C la liste des noms des élèves par classe

8. Un élève écrit SELECT nom, classe, COUNT(*) FROM Eleve GROUP BY classe pour compter les élèves par classe en affichant aussi leur nom. Cette requête est problématique car :

- ☐ A COUNT(*) ne peut jamais être utilisé avec GROUP BY
- ☐ B l'attribut nom n'est ni dans le GROUP BY ni agrégé, alors qu'il existe plusieurs noms par groupe
- ☐ C la clause FROM est absente

9. Écrire une requête donnant le nom des élèves de la classe 'Terminale NSI' nécessite d'utiliser la clause :

- ☐ A GROUP BY uniquement
- ☐ B une fonction d'agrégation uniquement
- ☐ C WHERE, pour filtrer sur la valeur de l'attribut classe

10. Le langage SQL sert à :

- ☐ A interroger et manipuler des données organisées en tables reliées entre elles
- ☐ B décrire uniquement la structure des tables, sans jamais interroger les données qu'elles contiennent
- ☐ C remplacer entièrement la notion de table et d'attribut



11

En 30 secondes — ta carte de dernière révision

découpe-la et garde-la dans ta trousse — révision de veille de contrôle !

- SELECT (attributs), FROM (table), WHERE (condition) : le squelette d'une requête SQL de sélection.
- Jointure SQL : égalité clé primaire / clé étrangère dans la clause WHERE, pour combiner deux tables.
- GROUP BY regroupe les n-uplets par attribut, une fonction d'agrégation (COUNT, SUM, AVG) calcule une valeur par groupe.

PLAN DE RÉVISION ESPACÉE — relis la carte et refais le QCM : ☐ J+1 ☐ J+3 ☐ J+7 ☐ J+14



QCM en ligne

VA PLUS LOIN — TESTE-TOI SUR CETTE NOTION

 Scanne le QR code (ou clique dessus) : le QCM chronométré de cette notion, note immédiate, chaque erreur expliquée. Gratuit, sans inscription.

 Pour t'entraîner — QCM en nombre, évaluations, explications détaillées, analyse de tes points faibles — l'abonnement Réussito (sans engagement) : via le QR code ou sur reussito.fr/abo

 Débloque toutes les fiches de Terminale : via le QR code ou sur reussito.fr/terminale



L'élève travaille sans les solutions sous les yeux ; vous corrigez ensemble avec ces pages. Chaque réponse est justifiée : pas besoin de connaître le cours.

Correction des exercices

Exercice 1. On affiche l'attribut nom (clause SELECT), depuis la table Eleve (clause FROM), en filtrant sur la condition classe = 'Terminale NSI' (clause WHERE) : `SELECT nom FROM Eleve WHERE classe = 'Terminale NSI'`.

Exercice 2. On regroupe les n-uplets par identifiant_eleve (clause GROUP BY), puisqu'on veut une moyenne par élève, et on calcule cette moyenne avec la fonction d'agrégation AVG sur l'attribut valeur : `SELECT identifiant_eleve, AVG(valeur) FROM Note GROUP BY identifiant_eleve`.

Réponses du QCM — expliquées une par une

- 1 → A** SELECT liste les attributs que la requête doit afficher. *Piège : table interrogée = décrit la clause FROM, pas SELECT ; condition de filtre = décrit la clause WHERE, pas SELECT.*
- 2 → B** FROM désigne la table (ou les tables) sur laquelle porte la requête. *Piège : attributs à afficher = décrit SELECT, pas FROM ; regroupement des n-uplets = décrit GROUP BY, pas FROM.*
- 3 → C** WHERE filtre les lignes selon une condition, en ne gardant que celles qui la vérifient. *Piège : afficher certains attributs = décrit SELECT, pas WHERE ; regrouper par valeur = décrit GROUP BY, pas WHERE.*
- 4 → A** La jointure s'appuie sur l'égalité entre clé primaire et clé étrangère pour relier les n-uplets des deux tables. *Piège : comparaison entre deux attributs de la même table = ne relie pas deux tables différentes ; aucune condition = sans condition, les tables ne sont pas reliées de façon cohérente.*
- 5 → B** Une fonction d'agrégation (COUNT, SUM, AVG...) résume un ensemble de n-uplets en une seule valeur. *Piège : nouvelle table ligne par ligne = ne correspond pas à une agrégation, qui résume au contraire les lignes ; le regroupement des n-uplets selon une valeur commune décrit la clause GROUP BY, qui prépare les groupes sur lesquels la fonction d'agrégation est ensuite appliquée — ce n'est pas la fonction d'agrégation elle-même qui effectue ce regroupement.*
- 6 → C** GROUP BY forme des groupes de n-uplets, sur lesquels une fonction d'agrégation est ensuite appliquée séparément. *Piège : filtrer avant regroupement = décrit la clause WHERE, pas GROUP BY ; afficher un seul attribut sans condition = ne décrit pas le rôle de regroupement de GROUP BY.*
- 7 → A** GROUP BY classe forme un groupe par classe, et COUNT(*) compte les n-uplets de chaque groupe. *Piège : nombre total toutes classes confondues = ignore le regroupement par classe introduit par GROUP BY ; liste des noms = nom n'est pas demandé dans cette requête, seul COUNT(*) est calculé.*
- 8 → B** Chaque attribut du SELECT doit être dans le GROUP BY ou agrégé ; nom ne l'est pas, ce qui rend le résultat ambigu. *Piège : COUNT(*) avec GROUP BY = c'est au contraire une combinaison courante et correcte ; clause FROM absente = FROM Eleve est bien présent dans cette requête.*
- 9 → C** Filtrer sur une valeur précise d'un attribut est le rôle de la clause WHERE. *Piège : GROUP BY uniquement = sert à regrouper, pas à filtrer une seule classe précise ; fonction d'agrégation uniquement = ne filtre aucun n-uplet, elle calcule une valeur résumée.*

10 → A

SQL est le langage standard pour écrire des requêtes sur des données organisées en tables reliées entre elles. *Piège : décrire uniquement la structure des tables sans jamais interroger les données = SQL sert aussi à interroger les données, pas seulement à décrire la structure des tables ; remplacer tables et attributs = SQL s'appuie au contraire sur ces notions, il ne les remplace pas.*