Bases de données

Master 1 Humanités numériques

2018-2019

Jérôme Darmont (en remplacement de Cécile Favre)
http://eric.univ-lyon2.fr/~cfavre/

1

Requêtes

- Quésaco ?

2

Pourquoi des requêtes

- Il est intéressant de pouvoir retrouver que certaines informations dans la base de données
- Lire tous les enregistrements peut être fastidieux
- Les requêtes vont donc permettre de n'afficher que les données qui nous intéressent

3

Trois types d'opérations

- Opérateurs relationnels de l'algèbre relationnelle
- Trois types :
  - Projection : on ne conserve que les champs intéressants
  - Sélection : on ne conserve que les enregistrements intéressants
  - Jointure : on remet ensemble des données situées dans différentes tables
- Possibilité de mélanger les opérations au sein d'une même requête

4

Projection / sélection

- Soit une table Etudiants(N°, nom, prénom, âge, année, établissement) qui contient 150 étudiant.es
- Projection : n'afficher que les noms et prénoms des étudiants
- Requête de sélection : n'afficher que les étudiants dont le nom commence par 'M'

N°	Nom	Prénom	Age	Année	Faculté
10	MERWEMT	Cécile	24	4	Sociologie
25	MUTARGO	Cédric	21	2	Histoire

Requête de sélection

Requête de projection

5

Requêtes de base

- Avec interface QBE

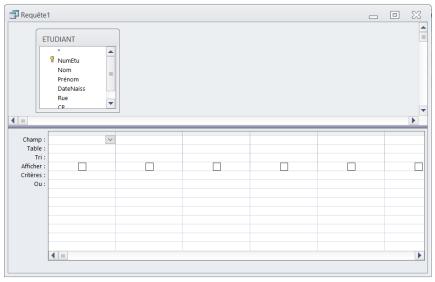
6

Effectuer une requête Access

- Menu Créer / Création de requête
- Ajouter les données servant de base à la requête (table(s), requête(s)…)
- Préciser la requête à travers l'interface

7

La fenêtre de requête interface QBE (Query By Example)



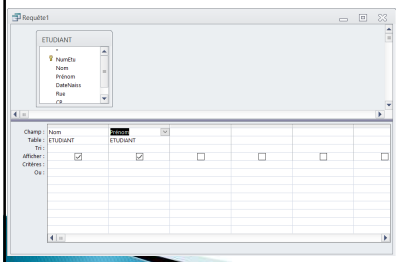
Les données de base

Partie requête

8

Faire une projection

- Il suffit de double-cliquer uniquement sur les champs souhaités



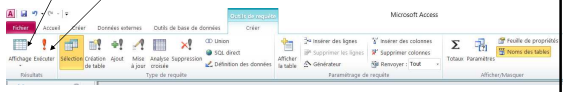
Seuls les champs sélectionnés s'ajoutent en bas. Ils seront les seuls à s'afficher (si la case « afficher » est bien cochée).

Le champ '*' signifie « tous les champs ».

9

Exécuter une requête

- Plusieurs possibilités :
  - Demander l'exécution
  - Changer le type d'affichage

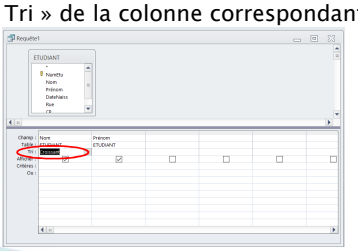


- Résultat : un ensemble d'enregistrements

10

Le tri

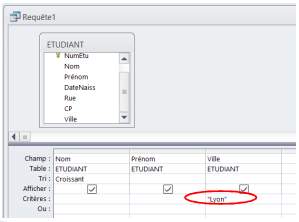
- Pour trier selon un ou plusieurs champs, il faut mettre « croissant » ou « décroissant » dans la case « Tri » de la colonne correspondante



11

Les critères : égalité

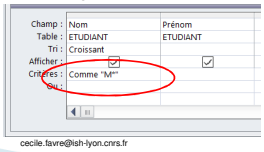
- Pour rechercher des informations quand on cherche une valeur en particulier
  - = valeur (numérique)
  - = "valeur" (texte)



12

Les critères : comme

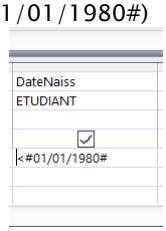
- A utiliser pour les champs de type « texte »
- Permet de dire à quoi doit ressembler le résultat
- Utilisation de deux caractères spéciaux :
  - * : n'importe quoi en nombre quelconque
  - ? : un caractère quelconque
- Ex : les noms qui commencent par 'M'



13

Les critères : comparaison

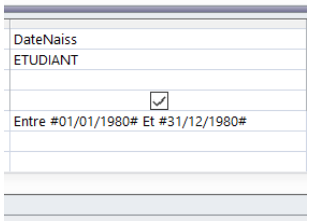
- Pour les dates et les nombres, utilisation des symboles de comparaison (>, <, >=, <=)
- Les dates sont encadrées de # (par exemple naissance avant #01/01/1980#)



14

Les critères : entre

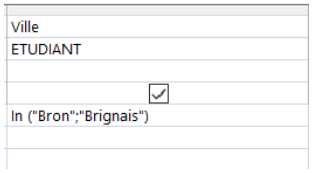
- Pour les dates et les nombres
- Possibilité d'indiquer deux bornes :
  Entre XXX et XXX



15

Les critères : liste

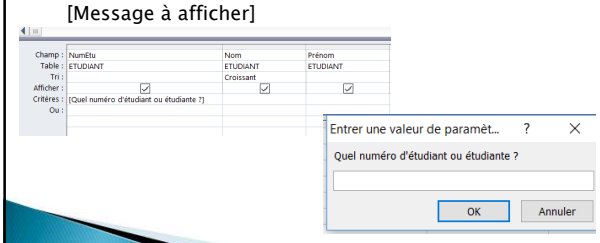
- Possibilité de préciser une liste de valeurs (avec ou sans " " selon type de données)
- Seuls les enregistrements correspondants à une des valeurs sont conservés



16

Requête paramétrée

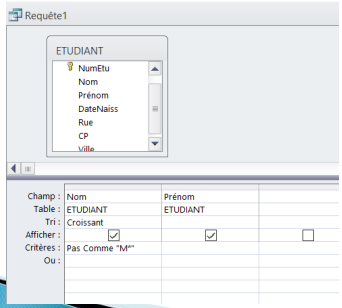
- On peut demander un critère à l'éxécution de la requête
- On met alors comme critère :
  [Message à afficher]



17

Négation de critère

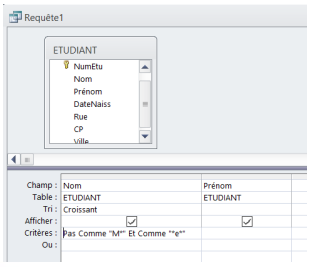
- On peut exclure des critères avec le Pas



18

Mettre plusieurs critères sur un champ

- On peut cumuler les critères pour un même champ avec « Et » et « Ou »



19

Mettre plusieurs critères sur différents champs

- Si les critères sont sur la même ligne : ET
- Si les critères sont sur des lignes différentes : OU

Champ :	Nom	Prénom	Ville
Table :	ETUDIANT	ETUDIANT	ETUDIANT
Tri :	Croissant		
Afficher :	☑	☑	☑
Critères :	Pas Comme "M*" Et Comme "*e*"		In ("Bron";"Brignais")
Ou :		Comme "P*"	

(Nom comme "M*" ET comme "*e*" **ET** Ville dans ("Bron";"Brignais"))
OU
Prénom comme "P*"

20

Recommandation

- Bien distinguer
  - restitution (case à cocher afficher)
  - critères

-> Possibilité de préciser des critères sans avoir le champ en question en restitution

- Tester en visualisant le résultat d'abord

21

Requêtes de base

Avec SQL

22

Présentation générale

- Les SGBD peuvent être interrogés via des commandes spécifiques
- Il existe un langage commun, normalisé, utilisé par la grande majorité des SGBD
- Ce langage s'appelle SQL pour « Structured Query Language »

23

Présentation générale

- SQL : définition, manipulation et contrôle d'une base de données relationnelle (basé sur l'algèbre relationnelle)
- SQL subdivisé en 3 sous langages :
  - LDD (Langage de Définition des Données) : création, modification et suppression définitions des tables
  - LMD (Langage de Manipulation de Données) : ajout, suppression, modification et interrogation des données
  - LCD (Langage de Contrôle de Données) : gestion des protections d'accès

24

Présentation générale

- SQL : définition, manipulation et contrôle d'une base de données relationnelle (basé sur l'algèbre relationnelle)
- SQL subdivisé en 3 sous langages :
  - LDD (Langage de Définition des Données) : création, modification et suppression définitions des tables
  - **LMD (Langage de Manipulation de Données) : ajout, suppression, modification et interrogation des données**
  - LCD (Langage de Contrôle de Données) : gestion des protections d'accès

25

SQL et Access

- Les requêtes faites dans Access via l'interface graphique sont en fait automatiquement traduites en SQL puis exécutées
- Par contre Access ne respecte pas tout à fait les standards SQL...

26

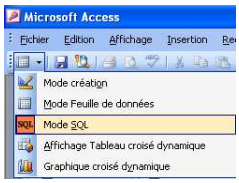
Les possibilités de SQL

- SQL permet de :
  - faire une requête sur les tables (requête avec projection, sélection, jointure)
  - créer/modifier/supprimer des tables ou des bases
  - ajouter/modifier/supprimer des enregistrements
  - gérer les droits
- Chaque instruction SQL se termine par « ; »

27

Faire du SQL dans Access

- Menu Créer / Création de requête
- Basculer en mode SQL
- Taper le code SQL
- Exécuter / enregistrer



28

Requête de base

- SELECT champs FROM tables;
- SELECT champs FROM tables WHERE predicats;
- Permet de conserver les « champs » des « tables » répondant aux « prédicats »
- * dans la clause SELECT signifie « tous les champs » (comme dans l'interface graphique)
- Ex :
  SELECT Nom, Prénom FROM Etudiant WHERE Ville='Lyon';
  SELECT * FROM Etudiant WHERE Ville='Lyon';

29

Les prédicats – 1

- > ; < ; <= ; >= ; = ; != : opérateurs de comparaison
- SELECT * FROM Etudiant WHERE Ville='Lyon';
  - Sélectionne l'ensemble des étudiant.es habitant Lyon
- SELECT * FROM Etudiant WHERE Nom='PERSONNE';
  - Sélectionne l'ensemble des étudiant.es ayant pour nom PERSONNE

30

Les prédicats – 2

- AND/OR : pour composer différents prédicats
- SELECT * FROM Etudiant WHERE Ville='Lyon' AND Nom='PERSONNE';
  - Sélectionne les étudiant.es habitant Lyon et dont le nom est PERSONNE

31

Les prédicats – 3

- LIKE : comparaison de chaînes (« comme »)
- SELECT * FROM Etudiant WHERE Adresse LIKE « %rue% »;
  - Sélectionne les étudiant.es qui habitent dans une rue
- Caractères joker :
  - % = correspond à * dans Access
  - _ = correspond à ? dans Access

32

Les prédicats – 4

- IN : présence dans une liste
- SELECT Nom, Prénom FROM Etudiant WHERE Ville IN ('Lyon', 'Bron', 'Brignais');
  - Sélectionne les étudiant.es qui habitent Lyon, Bron ou Brignais

33

Les prédicats – 5

- BETWEEN … AND … : pour donner une fourchette
- SELECT * FROM Etudiant WHERE Naissance BETWEEN #01/01/1980# AND #31/12/1980#;
  - Sélectionne les informations des étudiant.es né.es en 1980

34

Les prédicats – 6

- NOT : permet de faire la négation d'un prédicat
- SELECT * FROM Etudiant WHERE Ville NOT IN ('Lyon', 'Brignais');
  - Sélectionne les étudiant.es, sauf les personnes habitant Lyon ou Brignais

35

Autre clause : ORDER BY

- ORDER BY (champ) ASC/DESC : permet de trier les enregistrements
  - ASC : par ordre croissant ou alphabétique
  - DESC : par ordre décroissant
- Par défaut (sans précision) : ordre ascendant
- SELECT * FROM Etudiant ORDER BY Nom DESC;
  - Donne toutes les infos des étudiant.es triées par nom de famille (inversement à l'ordre alphabétique)

36

Requêtes calculs et regroupement

Avec interface QBE

37

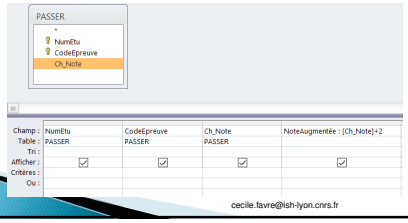
Les calculs dans les requêtes

- Pour certains calculs, il est inutile d'enregistrer toutes les données calculées
- On ne stocke donc que les données de base et les calculs seront refaits à chaque requête
- Compromis entre :
  - Place sur le disque nécessaire pour la base
  - Capacités de calcul de l'ordinateur

38

Faire un calcul sous Access

- Il suffit de mettre un nouveau champ
  - nom : le futur nom de la colonne
  - Suivi de deux points
  - Puis de la formule, avec les champs entre [ ]
- Exemple : NoteAugmentée : [Ch_Note] + 2



39

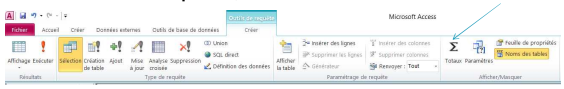
Les regroupements/agrégats

- Il est possible de regrouper certaines données selon un ou plusieurs critères
- On peut alors faire des calculs sur les autres champs : somme, moyenne, min, max, compte…
- Ex : moyenne des notes par étudiant
  - Regroupement par étudiant
  - Moyenne sur les notes

40

Faire un regroupement Access

- Cliquez sur le bouton Σ Totaux pour faire apparaître la ligne « Opérations » (Menu Outils de requête)

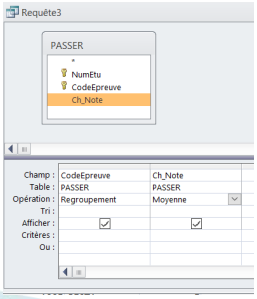


- Choisir :
  - Regroupement : pour les champs à regrouper
  - Les autres fonctions : pour les champs sur lesquels on fait des calculs (somme, moyenne…)
  - Où : lorsque l'on veut mettre un critère

41

Exemple

- Nombre d'étudiants par épreuve



42

Requêtes calculs et regroupement

Avec SQL

43

Rappels

- Requête simple :

SELECT champs FROM tables WHERE predicats;

- Permet de faire à la fois des projections (SELECT) et des sélections (WHERE)

44

Requête SELECT complète

- En réalité la requête SELECT contient beaucoup plus de clauses
- Voilà une version plus complète :

SELECT *champs*
FROM *tables*
WHERE *prédicats*
GROUP BY *critères de regroupements*
HAVING *critères sur le regroupement*
ORDER BY *champ de tri* ASC/DESC;

Seules clauses obligatoires

45

Clause GROUP BY

- Permet de regrouper les champs
- Correspond aux champs sous lesquels étaient écrits « regroupement » dans l'interface d'Access
- On peut mettre un ou plusieurs champs

46

Calculs sur les regroupements

- Pour faire des sommes, moyennes… il faut l'indiquer dans la clause SELECT
- Ex : pour faire la moyenne des notes :

```
SELECT NumEtu, avg(Ch_Note)
FROM Etudiant
GROUP BY NumEtu;
```


47

Calculs sur les groupements 2

- Voici quelques mots-clés pour les agrégats :
  - Sum() : somme
  - Avg() : moyenne
  - Min() : valeur minimale
  - Max() : valeur maximale
  - Count() : compte le nombre d'enregistrements

48

Renommage d'un champ

- Avec l'utilisation des groupements, il est souvent souhaitable d'avoir un nom plus court ou pour les calculs
- Dans la clause SELECT, on peut renommer un champ avec AS :

```
SELECT NumEtu, avg(Ch_Note) AS Moy
FROM Etudiant
GROUP BY NumEtu;
```


49

Calculs dans les requêtes

- Il suffit d'indiquer le calcul dans la clause SELECT
- Par exemple :

```
SELECT NumEtu, CodeEpreuve, Ch_Note + 2 AS
   NoteAugment
FROM Passer;
```

- L'utilisation du renommage est alors fortement conseillé pour un résultat clair

50

Clause HAVING

- Il s'agit de contraintes non pas sur les données d'origine mais sur les regroupements, portant sur des calculs d'agrégats (sum, avg…)
- Ne pas confondre WHERE et HAVING !

51

Requêtes de jointure

Avec interface QBE

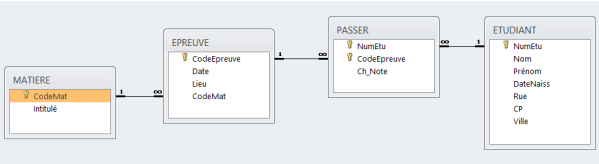
52

Requête sur une table

- Si on veut récupérer des informations sur une table, il suffit de faire une requête avec sélection / projection
- Problème : et si mes données sont dans deux tables différentes (ou plus) ???

53

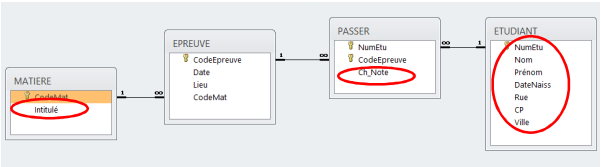
Exemple : base Etudiant



54

Exemple de questions

- Pour chaque étudiant.e, donner les informations sur les étudiant.es, les matières, et les notes obtenues.



55

Jointure

- Une jointure est une combinaison des enregistrements de deux ou plusieurs tables
- La combinaison se fait grâce aux liens clé primaire – clé étrangère

56

Jointure sous Access

- Sur la page « relations », il faut bien que les contraintes de clés étrangères aient été spécifiées
- Lors des requêtes, Access va utiliser ces informations pour réaliser les jointures

57

Procédure

- Choix des tables qui vont intervenir dans la requête + les tables qui les lient (toutes les tables choisies doivent être liées entre elles).
- Choix des champs et spécification des critères s'il y en a, comme dans toute requête. Le champ « table » correspond à la table d'où provient le champ.
- Lancement de la requête, Access fait la jointure.

58

ATTENTION

- Il faut bien mettre que des tables en relations dans les sources de la requête sinon Access fait un produit cartésien (toutes les possibilités)
- Il peut donc être nécessaire de rajouter des tables pour lier les tables « utiles » (celles qui contiennent les champs voulus)

59

Requêtes de jointure

» Avec SQL

60

Rappel

- Requête simple :

SELECT champs FROM tables WHERE predicats;

- Permet de faire à la fois des projections (SELECT) et des sélections (WHERE)

61

Jointures

- Il existe deux façons pour faire des jointures :
  - Utilisation du mot clé INNER JOIN
  - Utilisation de la clause WHERE
- Attention à ne pas confondre avec le produit cartésien

62

Jointure – solution 1

- On utilise la clause WHERE pour indiquer le champ de jointure
- Ex :
  - Etudiant(NumEtu, Nom, Prénom, DateNaiss, Rue, CP, Ville)
  - Passer(#NumEtu, #CodeEpreuve, Ch_Note)

```
SELECT NumEtu, Nom, Prénom, Ch_Note
FROM Etudiant, Passer
WHERE Etudiant.NumEtu = Passer.NumEtu;
```


63

Jointure – solution 2

- Cette solution utilise un mot-clé de SQL : INNER JOIN, qui se met dans FROM
- La requête précédente donne alors :

```
SELECT NumEtu, Nom, Prénom, Ch_Note
FROM Etudiant INNER JOIN Passer
ON Etudiant.NumEtu = Passer.NumEtu;
```


64

Jointure avec 3 tables

```
SELECT NumEtu, Nom, Prénom, CodeEpreuve, Date, Ch_Note
FROM Etudiant, Passer, Epreuve
WHERE Etudiant.NumEtu=Passer.NumEtu
AND Passer.CodeEpreuve=Epreuve.CodeEpreuve
AND Nom LIKE 'M*';

SELECT NumEtu, Nom, Prénom, CodeEpreuve, Date, Ch_Note
FROM (Etudiant INNER JOIN Passer
    ON Etudiant.NumEtu = Passer.NumEtu) INNER JOIN Epreuve
    ON Passer.CodeEpreuve=Epreuve.CodeEpreuve
WHERE Nom LIKE 'M*';
```


65

Jointure – comparaison

- La première solution est plus simple à écrire (surtout si + de 2 tables)
- Mais la deuxième a l'avantage d'être plus claire :
  - Dès la clause FROM on sait qu'on fait une jointure
  - La clause WHERE ne sert que pour les prédicats de sélection et pas pour la jointure
- Au final : à vous de choisir ce que vous préférez, mais vous devez savoir faire les deux

66

Autres Prédicats

- Prédicats EXISTS/NOT EXISTS
  - Dans la clause WHERE
  - Précision d'une requête
- Opérateurs ensemblistes :

INTERSECT, MINUS, UNION…

67

LDD

Définition des données

68

Tables

- CREATE TABLE nom_table (Attribut1 TYPE, Attribut2 TYPE, …, contrainte_intégrité1, contrainte_intégrité2, …);
- Principaux types de données :
  - NUMBER(n) : entier à n chiffres
  - NUMBER(n,m) : réel à n chiffres au total, virgule comprise, m chiffres après la virgule
  - VARCHAR(n) : chaîne de caractères (entre ' ')
  - DATE : date au format 'JJ-MM-AAAA'

69

Contraintes intégrité

- Clé primaire

CONSTRAINT nom_c PRIMARY KEY (attribut_clé)

- Clé étrangère

CONSTRAINT nom_c FOREIGN KEY (attribut_clé_etr) REFERENCES table(attribut)

- Contrainte de domaine

CONSTRAINT nom_c CHECK (condition)

70

Modifications structurelles

- Ajout d'attributs

ALTER TABLE nom_table ADD (attribut TYPE,…);

- Modifications d'attributs

ALTER TABLE nom_table MODIFY (attribut TYPE, …)

- Suppression d'attributs

ALTER TABLE nom_table DROP COLUMN attribut;

71

Modifications structurelles

- Ajout de contrainte

ALTER TABLE nom_table ADD CONSTRAINT nom_c déf_c;

- Suppression de contrainte

ALTER TABLE nom_table DROP CONSTRAINT nom_c;

72

Copie et destruction de tables

- Destruction

DROP TABLE nom_table;

- Copie

CREATE TABLE copie AS requete;

73

LMD

- Manipulation des données

(inclut les requêtes d'interrogation)

74

Insertion et mise à jour

- Ajout d'un n-uplet

INSERT INTO nom_table VALUES (val_att1, val_att2,…);

- Mise à jour d'un attribut

UPDATE nom_table SET attribut = valeur [WHERE condition];

75

Suppression

- Suppression de n-uplets

DELETE FROM nom_table [WHERE condition];

76

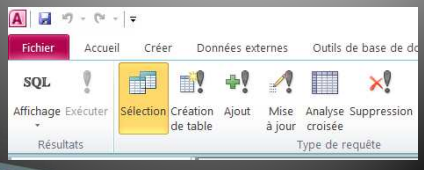
Interrogation des données

SELECT [ALL|DISTINCT] attribut(s) FROM table(s)
[WHERE condition]
[GROUP BY attribut(s) [HAVING condition]]
[ORDER BY attribut(s)];

77

Autres types de requête

- QBE



78

Requête création de table

- Sélection du type de requête
- Sélection des informations
- Préciser le nom de la table
- Exécuter la requête (ICI l'affichage permet de voir les données concernées)

79

Requête ajout de données

- Sélection du type de requête
- Sélection des informations et où elles doivent être ajoutées
- Exécuter la requête (ICI l'affichage permet de voir les données concernées)

80

Requête mise à jour

- Sélection du type de requête
- Sélection des informations et la mise à jour
- Exécuter la requête (ICI l'affichage permet de voir les données concernées)

81

Requête suppression

- Sélection du type de requête
- Sélection des informations Exécuter la requête (ICI l'affichage permet de voir les données concernées par la suppression)

82

Application

- Implémentation d'une base de données avec importation de données provenant d'Excel

83