BASE DE DONNÉES

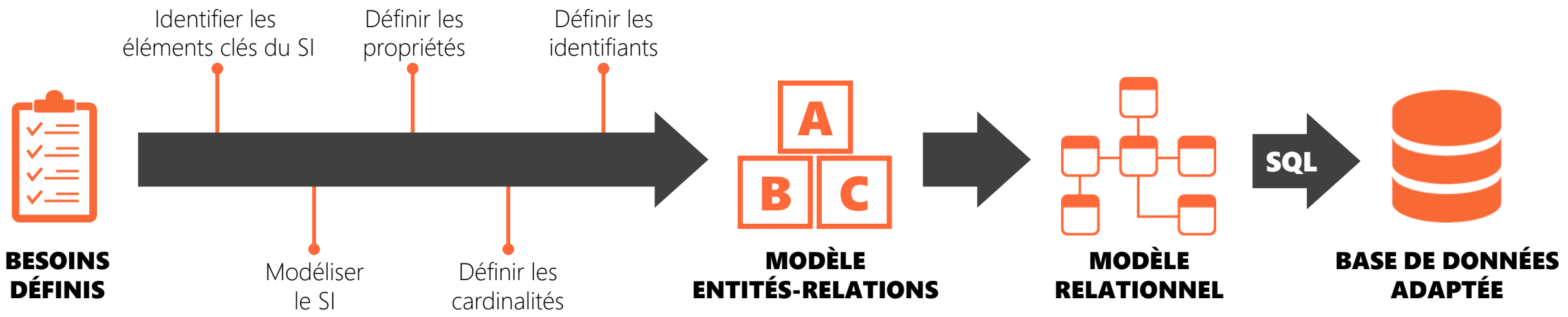
STRUCTURED QUERY LANGUAGE

charles.meunier@u-bourgogne.fr

IMPLÉMENTER UNE BASE DE DONNÉES

charles.meunier@u-bourgogne.fr

ET MAINTENANT ?



charles.meunier@u-bourgogne.fr

SQL

SQL (Structured Query Language) est un langage de programmation **DÉCLARATIF**.

Il permet de **CRÉER**, **MODIFIER** et **INTERROGER** des bases de données **RELATIONNELLES**

Langage **IMPÉRATIF**

Les instructions du programme **DÉCRIVENT LES ACTIONS À RÉALISÉES** pour obtenir le résultat souhaité

Langage **DÉCLARATIF**

Les instructions du programme **DÉCRIVENT LE RÉSULTAT SOUHAITÉ** sans indiquer comment y parvenir

charles.meunier@u-bourgogne.fr

GÉRER LES BASES DE DONNÉES

CRÉER UNE BASE DE DONNÉES

```
CREATE DATABASE nom_de_la_bdd;
```

SUPPRIMER UNE BASE DE DONNÉES

```
DROP DATABASE nom_de_la_bdd;
```

LISTER LES BASES DE DONNÉES

```
SHOW DATABASES;
```

TRAVAILLER AVEC UNE BASE DE DONNÉES

```
USE nom_de_la_bdd;
```

NOTE

Le point-virgule indique la fin de la requête. Si vous l'oublier, le SGBD estimera que la requête n'est pas terminée et attendra.

charles.meunier@u-bourgogne.fr

GÉRER LES TABLES

CRÉER UNE TABLE

```
CREATE TABLE `couleur` (
    `id` INT NOT NULL AUTO_INCREMENT,
    `nom` VARCHAR(50) NOT NULL,
    PRIMARY KEY(`id`)
);
```

Il est recommandé de délimiter les noms de tables et d'attributs avec des anti cotes (`) pour éviter qu'ils soient interprétés comme un mot clé SQL.

Exemple : `CREATE TABLE user;`
USER est un mot clé SQL. La requête provoquera une erreur.

SUPPRIMER UNE TABLE

```
DROP TABLE `couleur`;
```

MODIFIER UNE TABLE

```
ALTER TABLE `couleur`
ADD COLUMN `picture` BLOB NULL DEFAULT NULL;
```


charles.meunier@u-bourgogne.fr

TYPES DE DONNÉES

NUMÉRIQUE

- BOOL
- TINYINT
- INT
- BIGINT
- FLOAT
- DOUBLE
- DECIMAL
- ...

CHAINES DE CARACTÈRES

- CHAR(size)
- VARCHAR(size)
- TEXT
- LONGTEXT
- ...

TEMPOREL

- TIME
- DATE
- DATETIME
- TIMESTAMP
- YEAR
- ...

BINAIRE

- TINYBLOB
- BLOB
- MEDIUMBLOB
- LONGBLOB
- ...

DIVERS

- SET
- ENUM
- POLYGON
- JSON
- ...

ATTENTION : Le nom des types disponibles ainsi que leur taille en mémoire dépendent du SGBD utilisé. Pensez à lire la doc !

charles.meunier@u-bourgogne.fr

CLEF ÉTRANGÈRE

Carte (id, valeur, retournée, *idCouleur*, *idPile*, index)

```
CREATE TABLE `carte` (
    `id` INT NOT NULL AUTO_INCREMENT,
    `valeur` INT NOT NULL,
    `retournee` BOOL NOT NULL DEFAULT 0,
    `idCouleur` INT NOT NULL,
    `idPile` INT NOT NULL,
    `index` INT NOT NULL,
    PRIMARY KEY(`id`)
);
```

```
ALTER TABLE `carte`
ADD CONSTRAINT `FK_carte_couleur`
FOREIGN KEY (`idCouleur`)
REFERENCES `couleur` (`id`)
ON UPDATE NO ACTION
ON DELETE NO ACTION;
```

ADD CONSTRAINT

Ajout d'une contrainte d'intégrité permettant au SGBD de s'assurer que la valeur fournie pour idCouleur correspond à un id de la table Couleur.

FOREIGN KEY

Indique le ou les attributs participants à la contrainte de clé étrangère.

REFERENCES

Définit à quels attributs de quelle table source est liée la clé étrangère.

ON UPDATE

Que faire si la valeur de la clé étrangère est mise à jour dans la table source ?

ON DELETE

Que faire si l'occurrence à laquelle est lié la clé étrangère est supprimée dans la table source ?

charles.meunier@u-bourgogne.fr

CLEF ÉTRANGÈRE - ON UPDATE / ON DELETE

Les options **ON UPDATE** et **ON DELETE** définissent le comportement à adopter en cas de modification ou de suppression de la valeur de la clé étrangère **DANS LA TABLE PARENTE**.

	ON UPDATE	ON DELETE
CASCADE	Les valeurs des clés étrangères sont automatiquement mises à jour dans la table enfant	La suppression d'une occurrence de la table parent entraîne la suppression des occurrences de la table enfant qui lui sont liées.
SET NULL	Les valeurs des clés étrangères sont définies à NULL dans la table enfant	Les valeurs des clés étrangères sont définies à NULL dans la table enfant
RESTRICT	Empêche la mise à jour de la valeur de la clé étrangère dans la table parente	Empêche la suppression de l'occurrence dans la table parente
NO ACTION	Idem RESTRICT	Idem RESTRICT

charles.meunier@u-bourgogne.fr

MANIPULATION DES DONNÉES

charles.meunier@u-bourgogne.fr

CRUD

CREATE

```
INSERT INTO `couleur` (nom)
VALUES ('Carreau'), ('Coeur'),
('trefle'), ('Pique');
```

On indique après le nom de la table la liste des attributs qui seront présents dans les tuples et dans quel ordre ils seront mentionnés.

Ici, l'attribut **id** n'est pas indiqué car sa valeur est générée automatiquement par le SGBD (**AUTO_INCREMENT**)

Le mot clé **VALUES** est suivi des tuples représentant les données à insérer

UPDATE

```
UPDATE `couleur`
SET nom = 'Trèfle'
WHERE nom = 'trefle';
```

La clause **SET** permet de lister les attributs à modifier ainsi que la nouvelle valeur à affecter.

La clause **WHERE** permet de limiter les modifications aux occurrences qui respectent la condition présentée.

Ici, seules les occurrences ayant la valeur **'trefle'** pour l'attribut **nom** seront modifiées.

DELETE

```
DELETE FROM `couleur`
WHERE nom = 'Trèfle';
```

La clause **WHERE** permet de ne supprimer que les occurrences qui respectent la condition présentée.

Ici, seules les occurrences ayant la valeur **'Trèfle'** pour l'attribut **nom** seront modifiées.

ATTENTION : EN CAS D'OUBLI DE LA CLAUSE WHERE, TOUTES LES DONNÉES DE LA TABLE SERONT MODIFIÉES/SUPPRIMÉES. IL N'Y A PAS DE RETOUR EN ARRIÈRE POSSIBLE.

charles.meunier@u-bourgogne.fr

SELECT

COMMENT AFFICHER LA LISTE DES FIGURES QUI SONT ACTUELLEMENT FACE CACHÉE DANS LE JEU ?

```sql
SELECT `valeur`, `idCouleur`
FROM `carte`
WHERE
    retournee = 0
    AND valeur > 10
ORDER BY
    idCouleur ASC,
    valeur ASC;
```

SELECT

Instruction permettant de récupérer des données. Elle est suivie de la liste des colonnes de la table que l'on souhaite obtenir.

FROM

De quelle table doit-on extraire les informations ?

WHERE

Définit des critères de filtrage pour affiner la recherche. Les opérateurs AND et OR permettent de construire des expressions booléennes.

ORDER BY

Permet de trier les résultats selon les valeurs d'une ou plusieurs colonnes. ASC précise que le tri est croissant, DESC que le tri est décroissant.

charles.meunier@u-bourgogne.fr

FONCTIONS SQL

QUELLE VALEUR OBTIENDRONS-NOUS SI NOUS ADDITIONNIONS LES VALEURS DE TOUTES LES CARTES DU JEU ?

```
SELECT SUM(`valeur`) AS `total`
FROM `carte`;
```

Combien y a-t-il de cartes dans le JEU ?

```
SELECT COUNT(*) AS `nombreCartes`
FROM `carte`;
```

SUM, COUNT, ...

SQL fournit un ensemble de fonctions arithmétiques, de manipulation de chaines de caractères (`CONCAT`, `PASSWORD`, ...), de gestion de dates (`NOW`, ...), ...

AS

Permet de redéfinir le nom d'une colonne dans l'affichage des résultats de la requête.

*

`SELECT *` indique que toutes les colonnes des tables impliquées dans les requêtes seront affichées.

charles.meunier@u-bourgogne.fr

GROUP BY

COMBIEN Y A-T-IL DE CARTES RETOURNÉES PAR COULEUR ?

```
SELECT COUNT(*) AS `nombre`, `idCouleur`
FROM `carte`
WHERE
    `retournee` = 1
GROUP BY
    `idCouleur`;
```

1. SELECTIONNE LES LIGNES CORREPSONDANT À LA REQUÊTE

id	valeur	idCouleur	retournee
5	5	1	1
9	9	1	1
18	3	2	1
27	12	3	1
54	7	2	1
56	10	1	1

2. REGROUPE LES LIGNES EN FONCTION DE LA VALEUR DE IDCOULEUR

id	valeur	idCouleur	retournee
5	5	1	1
9	9	1	1
56	10	1	1
18	3	2	1
54	7	2	1
27	12	3	1

3. APPELLE LA FONCTION COUNT POUR CHAQUE GROUPE DE LIGNES

nombre	idCouleur
3	1
2	2
1	3

charles.meunier@u-bourgogne.fr

GROUP BY

AFFICHER LA SOMME DES VALEURS DES CARTES RETOURNÉES DE CHAQUE COULEUR, SI CETTE SOMME EST SUPÉRIEURE À 20

```
SELECT SUM(`valeur`) AS `total`,
`idCouleur`
FROM `carte`
WHERE
    `retournee` = 1
GROUP BY
    `idCouleur`
HAVING
    `total` > 20;
```

HAVING

Applique un filtre sur le résultat des fonctions utilisées dans la clause SELECT telles que SUM, COUNT, AVG, MIN, ...

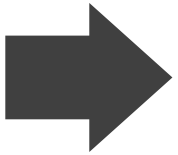
HAVING est souvent utilisé à la suite d'une clause GROUP BY.

charles.meunier@u-bourgogne.fr

JOINTURES

QUELLES SONT LES CARTES SITUÉES DANS LA PIOCHE ?

```
SELECT `valeur`, `idCouleur`
FROM `carte`
WHERE
    idPile = 1;
```



valeur	idCouleur
11	1
2	2

Résultat difficilement exploitable. Ce serait bien de pouvoir afficher à l'utilisateur le nom des couleurs plutôt que leur id en base de données.

charles.meunier@u-bourgogne.fr

JOINTURES

QUELLES SONT LES CARTES SITUÉES DANS LA PIOCHE ?

```
SELECT `carte`.`valeur`, `couleur`.`nom`
FROM `carte`
INNER JOIN `couleur`
    ON `couleur`.`id` = `carte`.`idCouleur`
WHERE
    idPile = 1;
```

CARTES SUR LA PILE

valeur	idCouleur
11	1
2	2

COULEURS

id	nom
1	Carreau
2	Cœur
3	Trèfle
4	Pique

PRODUIT CARTÉSIEN

valeur	idCouleur	id	nom
11	1	1	Carreau
11	1	2	Cœur
11	1	3	Trèfle
11	1	4	Pique
2	2	1	Carreau
2	2	2	Cœur
2	2	3	Trèfle
2	2	4	Pique

RÉSULTAT DE LA JOINTURE

valeur	nom
11	Carreau
2	Cœur

INNER JOIN

Réalise le produit cartésien de deux tables. Chaque occurrence de la table 1 est associée à toutes les occurrences de la table 2.

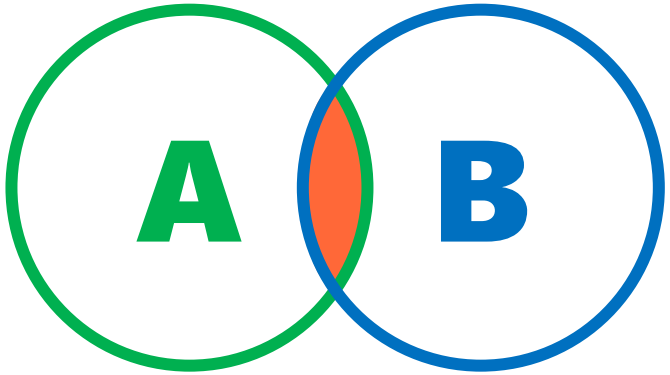
ON

Indique les attributs à associer de manière à ce que la jointure fournisse un résultat cohérent.

charles.meunier@u-bourgogne.fr

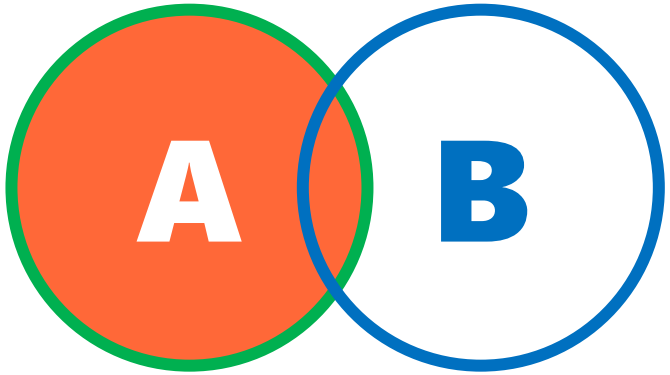
JOINTURES

INNER JOIN



Seuls les résultats capables d'associer des occurrences des deux tables seront présentés.

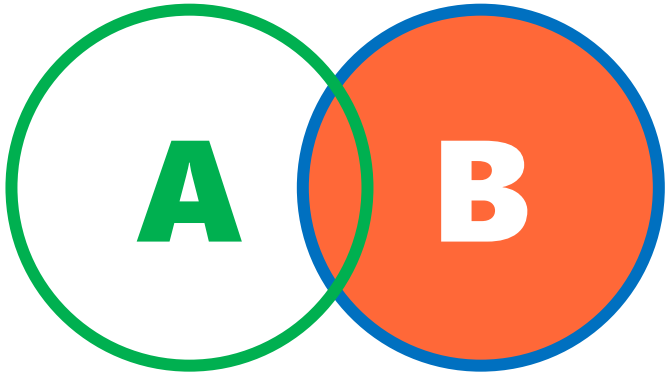
LEFT JOIN



Toutes les occurrences **A** sont combinées aux lignes correspondantes de **B**. S'il n'y a pas de correspondance, les colonnes de **B** sont présentée avec la valeur `NULL`.

Toutes les occurrences de **A** seront présentes de les résultats.

RIGHT JOIN



Toutes les occurrences **B** sont combinées aux lignes correspondantes de **A**. S'il n'y a pas de correspondance, les colonnes de A sont présentée avec la valeur `NULL`.

Toutes les occurrences de **B** seront présentes de les résultats.

charles.meunier@u-bourgogne.fr

LOGICIELS

charles.meunier@u-bourgogne.fr

MARIADB

TÉLÉCHARGEMENT DE MARIADB

https://mariadb.org/download

Disponible sur Windows, MacOS et Linux

MÉMORISEZ BIEN VOTRE MOT DE PASSE ADMINISTRATEUR !

charles.meunier@u-bourgogne.fr

MARIADB

INSTALLATION SOUS LINUX

```
# Installation
> sudo apt update
> sudo apt install mariadb-server

# Configuration - MariaDB est un fork de MySQL donc c'est normal :)
> sudo mysql_secure_installation
```

MÉMORISEZ BIEN VOTRE MOT DE PASSE ADMINISTRATEUR !

charles.meunier@u-bourgogne.fr

JUSTE AU CAS OÙ...

charles.meunier@u-bourgogne.fr

MARIADB

CONNEXION

```
> mariadb -u [username] -p
```

-u : utilisateur avec lequel vous souhaitez vous connecter

-p : demander le mot de passe à la connexion (vous savez, celui qu'il ne faut pas oublier...)

charles.meunier@u-bourgogne.fr

MANAGERS

MYSQL WORKBENCH

https://dev.mysql.com/downloads/workbench/

HEIDISQL

https://www.heidisql.com/

charles.meunier@u-bourgogne.fr