

Bases de données relationnelles

G. Dewaele

Les bases de données

L'un des objectifs de l'informatique est le traitement de grandes quantités d'informations.

Une solution efficace pour organiser une grande quantité d'informations est la *base de données relationnelle*.

Les outils permettant d'interagir avec ces bases de données sont appelés *systèmes de gestion de bases de données (SGBD)*.

Il en existe de nombreux, beaucoup d'entre eux utilisant, pour interagir avec les données, le langage *SQL (Simple Query Language)*.

Exemples : MySQL, Oracle, PostgreSQL, SQLite...

Description générale : entités, associations...

Couche « logique » : tables

Couche matérielle : stockage en mémoire, sur disques, distribuée sur le réseau...

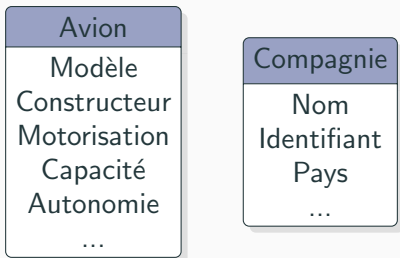
Type entité : description d'éléments de même nature (entités)

Par exemple, pour une base de données aéronautiques :

- modèles d'avions
- compagnies d'aviation
- pilotes
- trajets
- pièces détachées
- fabricants...

Le détail dépend fortement de l'usage !

Les types entité définissent des *attributs*



Les attributs ont un *domaine* (valeurs possibles)

Les entités associent des *valeurs* à ces attributs

Il existe des relations entre les différentes entités

Par exemple, quelle compagnie possède quels modèles d'avions

Nous y reviendrons en détail plus tard...

Au niveau logique, une *base de données* range les entités d'un type entité dans une *table*, assimilable à un tableau bi-dimensionnel.

TABLE Avions :

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Représentation logique en table

mais aucune hypothèse sur le stockage physique

On recherche performances, cohérence (pas de redondance), tolérance aux pannes...

Une base de données comprend généralement plus d'une table, et les systèmes de gestion de base de données permettent de *croiser* les données de plusieurs tables.

TABLE Compagnies :

| Code | Compagnie | Modèle | Nombre |
|------|------------|----------|--------|
| AF | Air France | 777-200 | 25 |
| AF | Air France | A321-200 | 15 |
| AF | Air France | A330-200 | 15 |
| AF | Air France | A340-300 | 12 |
| AC | Air Canada | 777-200 | 6 |
| AC | Air Canada | A321-200 | 10 |
| AC | Air Canada | CRJ-100 | 51 |

Attributs et entités dans une table

Chaque colonne d'une table correspond à un *attribut*.

Chaque entité correspond à une ligne dans la table.

TABLE Avions :

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Attributs manquants

Certaines « cases » d'une table peuvent être vides.

(En pratique, elles contiennent une valeur spéciale, NULL, ne faisant pas partie du domaine de l'attribut en question).

TABLE Avions :

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | | 1574 |

Objectifs d'une base de données

Les bases de données permettent de stocker une grande quantité de données de différentes natures, et de les mettre en relation afin de répondre à des questions variées et complexes :

Quel est le bi-moteur encore en service avec la plus grande capacité ?

Combien de modèles d'Airbus contient la flotte d'Air France ?

Quels sont les fabricants d'avions qui proposent un modèle permettant d'effectuer Paris - Pékin sans escale ?

Quelle est la dix-septième compagnie dont le nom ne commence pas par « A » ayant le plus d'avions pouvant transporter entre quatre-vingt-sept et cent onze passagers ?

Et globalement, toutes ces questions qui vous empêchent de dormir la nuit.

Principe des requêtes

Le langage SQL permet de transcrire ces questions sous la forme de *requêtes* qui pourront être comprises par la base de donnée.

Le résultat d'une requête **doit impérativement avoir la forme d'une table** (qui peut-être réduite à une seule case).

Quels sont les fabricants d'avions ont fourni des appareils biréacteurs actuellement en service dans la flotte d'Air France ?

```
SELECT DISTINCT A.Fabricant
FROM Avions AS A JOIN Compagnies AS C
ON A.Modèle = C.Modèle
WHERE A.Moteurs = 2 AND C.Compagnie = 'Air France'
```

On peut sélectionner certains attributs en effectuant une *projection* grâce au mot-clé SELECT :

SELECT Modèle, Moteurs FROM Avions

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Suppression des doublons

Le mot-clé DISTINCT, placé derrière le mot-clé SELECT, permet d'éliminer les doublons dans le résultat :

SELECT DISTINCT Fabricant, Moteurs FROM Avions

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Il est possible de créer temporairement des attributs supplémentaires en utilisant des expressions :

SELECT Modèle, Places/Moteurs FROM Avions

| Modèle | Fabricant | Moteurs | Places | Autonomie | Pl./Mot. |
|----------|------------|---------|--------|-----------|----------|
| 737-900 | Boeing | 2 | 215 | 5926 | 107.5 |
| 777-200 | Boeing | 2 | 440 | 14307 | 220 |
| A321-200 | Airbus | 2 | 220 | 5926 | 110 |
| A330-200 | Airbus | 2 | 380 | 13890 | 190 |
| A340-300 | Airbus | 4 | 440 | 13705 | 110 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | 25 |

Sélection sur critère booléen

On peut également effectuer des *sélections*, c'est-à-dire ne conserver que les lignes vérifiant une expression booléenne grâce au mot-clé WHERE :

```
SELECT Modèle, Fabricant, Moteurs, Places, Autonomie FROM Avions  
WHERE Places > 250
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Sélection sur critère booléen

Pour obtenir la totalité des attributs, on peut utiliser « * » :

```
SELECT * FROM Avions
```

```
WHERE Places > 250
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Expressions booléennes utiles

Pour savoir si un attribut figure parmi un ensemble de valeurs, on dispose du mot-clé IN :

```
SELECT * FROM table WHERE nom IN (val1, val2, val3)
```

Pour identifier les « cases vides » (valeur manquante d'un attribut dans une ligne), on utilisera IS NULL (ou IS NOT NULL) :

```
SELECT * FROM table WHERE nom IS NOT NULL
```

Pour faire des recherches sur une partie d'une chaîne, on peut utiliser LIKE, le caractère % pouvant remplacer n'importe quelle suite de caractères :

```
SELECT * FROM table WHERE nom LIKE '% Dupont'
```

Opérateurs booléens

Les opérateurs booléens OR, AND et NOT permettent de combiner les expressions booléennes :

```
SELECT * FROM Avions
```

```
WHERE Places > 250 AND Moteurs = 2
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Il est bien entendu possible de combiner sélection et projection :

```
SELECT Modèle, Moteurs FROM Avions
```

```
WHERE Places > 250 AND Moteurs = 2
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Tri des résultats

Pour ordonner les résultats, on peut utiliser le mot-clé ORDER BY :

(on peut utiliser une expression, et les attributs n'ont pas besoin d'être sélectionnés)

```
SELECT Modèle, Moteurs FROM Avions
```

```
ORDER BY Places
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| CRJ-100 | Bombardier | 2 | 50 | 1574 |
| 737-900 | Boeing | 2 | 215 | 5926 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A340-300 | Airbus | 4 | 440 | 13705 |

Le mot-clé DESC permet d'obtenir les résultats dans l'ordre décroissant :

```
SELECT Modèle, Moteurs FROM Avions
```

```
ORDER BY Places DESC
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 777-200 | Boeing | 2 | 440 | 14307 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| 737-900 | Boeing | 2 | 215 | 5926 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Limitation du nombre de résultats

On peut ne conserver que les n premiers résultats avec le mot-clé LIMIT :

```
SELECT Modèle, Moteurs FROM Avions
```

```
ORDER BY Places
```

```
LIMIT 3
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| CRJ-100 | Bombardier | 2 | 50 | 1574 |
| 737-900 | Boeing | 2 | 215 | 5926 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A340-300 | Airbus | 4 | 440 | 13705 |

On peut ignorer les n premiers résultats avec le mot-clé OFFSET :

```
SELECT Modèle, Moteurs FROM Avions
```

```
ORDER BY Places
```

```
LIMIT 3 OFFSET 2
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| CRJ-100 | Bombardier | 2 | 50 | 1574 |
| 737-900 | Boeing | 2 | 215 | 5926 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A340-300 | Airbus | 4 | 440 | 13705 |

La fonction COUNT permet d'effectuer des décomptes dans le résultat de la requête :

```
SELECT COUNT(*) FROM Avions
```

| Modèle | Fabricant | Moteurs | Places | Autonomie | COUNT |
|----------|------------|---------|--------|-----------|-------|
| 737-900 | Boeing | 2 | 215 | 5926 | 6 |
| 777-200 | Boeing | 2 | 440 | 14307 | |
| A321-200 | Airbus | 2 | 220 | 5926 | |
| A330-200 | Airbus | 2 | 380 | 13890 | |
| A340-300 | Airbus | 4 | 440 | 13705 | |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | |

Les « cases » vides (contenant NULL) ne sont pas comptées.

COUNT(Moteurs) compte les lignes pour lesquels l'attribut Moteur n'est pas vide.

COUNT(Moteurs, Places) compte les lignes pour lesquels les attributs Moteur et Places ne sont pas *tous les deux* vides.

COUNT(*) compte toutes les lignes.

Compter les valeurs uniques

Il est possible de dénombrer les différentes valeurs prises par l'attribut, grâce à DISTINCT :

```
SELECT COUNT(DISTINCT Fabricant) FROM Avions
```

| Modèle | Fabricant | Moteurs | Places | Autonomie | COUNT |
|----------|------------|---------|--------|-----------|-------|
| 737-900 | Boeing | 2 | 215 | 5926 | 3 |
| 777-200 | Boeing | 2 | 440 | 14307 | |
| A321-200 | Airbus | 2 | 220 | 5926 | |
| A330-200 | Airbus | 2 | 380 | 13890 | |
| A340-300 | Airbus | 4 | 440 | 13705 | |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | |

Si l'on combine une sélection et un décompte, seuls sont comptés les lignes respectant la condition :

```
SELECT COUNT(*) FROM Avions  
WHERE Moteurs = 2
```

| Modèle | Fabricant | Moteurs | Places | Autonomie | COUNT |
|----------|------------|---------|--------|-----------|-------|
| 737-900 | Boeing | 2 | 215 | 5926 | 5 |
| 777-200 | Boeing | 2 | 440 | 14307 | |
| A321-200 | Airbus | 2 | 220 | 5926 | |
| A330-200 | Airbus | 2 | 380 | 13890 | |
| A340-300 | Airbus | 4 | 440 | 13705 | |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | |

Il est également possible de déterminer la plus grande valeur d'un attribut grâce au mot-clé MAX :

```
SELECT MAX(Places) FROM Avions  
WHERE Autonomie < 10000
```

| Modèle | Fabricant | Moteurs | Places | Autonomie | MAX(Pl.) |
|----------|------------|---------|--------|-----------|----------|
| 737-900 | Boeing | 2 | 215 | 5926 | 220 |
| 777-200 | Boeing | 2 | 440 | 14307 | |
| A321-200 | Airbus | 2 | 220 | 5926 | |
| A330-200 | Airbus | 2 | 380 | 13890 | |
| A340-300 | Airbus | 4 | 440 | 13705 | |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | |

Diverses fonctions sont disponibles :

- MAX : retourne le plus grand élément
- MIN : retourne le plus petit élément
- SUM : retourne la somme des éléments
- AVG : retourne la moyenne des éléments

...

La disponibilité de ces fonctions peut varier d'un système de gestion de bases de données à l'autre.

AVG(x) peut être remplacé par $SUM(x)/COUNT(*)$

Mélanger attributs et fonctions

Le résultat devant être une table, mélanger attributs et fonctions peut être illégal.

La requête suivante est n'est pas correcte en SQL :

```
SELECT Modèle, MAX(Places) FROM Avions
```

| Modèle | Fabricant | Moteurs | Places | Autonomie | MAX(Pl.) |
|----------|------------|---------|--------|-----------|----------|
| 737-900 | Boeing | 2 | 215 | 5926 | 440 |
| 777-200 | Boeing | 2 | 440 | 14307 | |
| A321-200 | Airbus | 2 | 220 | 5926 | |
| A330-200 | Airbus | 2 | 380 | 13890 | |
| A340-300 | Airbus | 4 | 440 | 13705 | |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | |

Certains SQBD peuvent ne pas suivre la norme et retourner quelque chose (ex : SQLite), parfois ce qu'on peut attendre, parfois n'importe quoi.

```
SELECT Modèle, MAX(Places) FROM Avions
```

 peut avoir du sens

```
Mais SELECT Modèle, MIN(Places), MAX(Places) FROM Avions ?!
```

Il est possible de regrouper les valeurs identiques d'un attribut, grâce au mot-clé GROUP BY :

```
SELECT Fabricant FROM Avions
```

```
GROUP BY Fabricant
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | | 2 | 380 | 13890 |
| A340-300 | | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Le résultat doit rester une table, aussi sélectionner des attributs non-groupés ne fonctionne pas toujours :

```
SELECT Modèle, Fabricant FROM Avions
```

```
GROUP BY Fabricant
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | | 2 | 380 | 13890 |
| A340-300 | | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

COUNT, MIN, MAX... et groupes

Les fonctions telles que COUNT ou MAX agissent sur les groupes :

```
SELECT Fabricant, COUNT(*) FROM Avions
```

GROUP BY Fabricant

| Modèle | Fabricant | Moteurs | Places | Autonomie | COUNT |
|----------|------------|---------|--------|-----------|-------|
| 737-900 | Boeing | 2 | 215 | 5926 | 2 |
| 777-200 | | 2 | 440 | 14307 | |
| A321-200 | Airbus | 2 | 220 | 5926 | 3 |
| A330-200 | | 2 | 380 | 13890 | |
| A340-300 | | 4 | 440 | 13705 | |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | 1 |

Groupes sur plusieurs critères

Il est possible de regrouper les lignes sur plusieurs attributs :

```
SELECT Fabricant, MIN(Places) FROM Avions
```

```
GROUP BY Fabricant, Moteurs
```

| Modèle | Fabricant | Moteurs | Places | Autonomie | MIN(Pl.) |
|----------|------------|---------|--------|-----------|----------|
| 737-900 | Boeing | 2 | 215 | 5926 | 215 |
| 777-200 | | | 440 | 14307 | |
| A321-200 | Airbus | 2 | 220 | 5926 | 220 |
| A330-200 | | | 380 | 13890 | |
| A340-300 | Airbus | 4 | 440 | 13705 | 440 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | 50 |

Filtrer les groupes

De la même façon que l'on peut sélectionner des lignes, on peut filtrer les groupes grâce au mot-clé **HAVING** :

```
SELECT Fabricant, MIN(Places) FROM Avions  
GROUP BY Fabricant, Moteurs  
HAVING MIN(Places) > 215
```

| Modèle | Fabricant | Moteurs | Places | Autonomie | MIN(Pl.) |
|----------|------------|---------|--------|-----------|----------|
| 737-900 | Boeing | 2 | 215 | 5926 | 215 |
| 777-200 | | | 440 | 14307 | |
| A321-200 | Airbus | 2 | 220 | 5926 | 220 |
| A330-200 | | | 380 | 13890 | |
| A340-300 | Airbus | 4 | 440 | 13705 | 440 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | 50 |

Double filtrage

WHERE et HAVING peuvent être utilisés dans une même requête (WHERE agit avant le regroupement, HAVING après) :

```
SELECT Fabricant, MIN(Places) FROM Avions
```

```
WHERE Autonomie > 10000
```

```
GROUP BY Fabricant, Moteurs
```

```
HAVING MIN(Places) > 215
```

| Modèle | Fabricant | Moteurs | Places | Autonomie | MIN(Pl.) |
|----------|------------|---------|--------|-----------|----------|
| 737-900 | Boeing | 2 | 215 | 5926 | 440 |
| 777-200 | | | 440 | 14307 | |
| A321-200 | Airbus | 2 | 220 | 5926 | 380 |
| A330-200 | | | 380 | 13890 | |
| A340-300 | Airbus | 4 | 440 | 13705 | 440 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | 50 |

Il convient de conserver les éléments de la requête dans un ordre logique, correspondant à la chronologie des opérations :

- WHERE avant GROUP BY
- HAVING après GROUP BY
- ORDER BY après les opérations de sélection
- LIMIT / OFFSET en dernier

La projection (SELECT) est la dernière opération effectuée, elle n'est placée en tête que par commodité pour la lecture.

Exemple de requête combinant tous les éléments

```
SELECT Fabricant, MIN(Places) FROM Avions
```

```
WHERE Autonomie > 10000
```

```
GROUP BY Fabricant, Moteurs
```

```
HAVING MIN(Places) > 215
```

```
ORDER BY MIN(Places) DESC
```

```
LIMIT 1 OFFSET 2
```

| Modèle | Fabricant | Moteurs | Places | Autonomie | MIN(Pl.) |
|----------|------------|---------|--------|-----------|----------|
| 737-900 | Boeing | 2 | 215 | 5926 | 440 |
| 777-200 | | | 440 | 14307 | |
| A321-200 | Airbus | 2 | 220 | 5926 | 380 |
| A330-200 | | | 380 | 13890 | |
| A340-300 | Airbus | 4 | 440 | 13705 | 440 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | 50 |

Sous-requêtes

Le résultat d'une requête peut servir dans une autre requête. On place la sous-requête entre parenthèses.

Pour une sous-requête renvoyant une valeur, dans une expression :

```
SELECT * FROM Avions
```

```
WHERE Places > (SELECT AVG(Places) FROM Avions) :
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Sous-requêtes

Pour une sous-requête renvoyant une « liste » (table réduite à un seul attribut), dans une expression booléenne, avec le mot-clé IN :

```
SELECT Modèles FROM Avions
```

```
WHERE Fabricant IN
```

```
(SELECT DISTINCT Fabricant FROM Avions
```

```
WHERE Places > 400)
```

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Sous-requêtes

Enfin, une sous-requête renvoyant une « table » peut servir de table :

```
SELECT AVG(TMin) FROM  
    (SELECT Fabricant, MIN(Places) AS TMin  
     FROM Avions GROUP BY Fabricant)
```

| Modèle | Fabricant | Mot. | Places | Auton. | TMin | AVG(TM.) |
|----------|------------|------|--------|--------|------|----------|
| 737-900 | Boeing | 2 | 215 | 5926 | 215 | 161.33 |
| 777-200 | | 2 | 440 | 14307 | | |
| A321-200 | Airbus | 2 | 220 | 5926 | 220 | |
| A330-200 | | 2 | 380 | 13890 | | |
| A340-300 | | 4 | 440 | 13705 | | |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | 50 | |

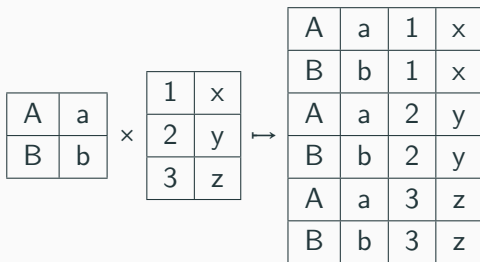
On remarquera le AS pour nommer une colonne de la sous-requête.

Combiner les tables

Le plein intérêt des systèmes de gestion de bases de données ne se révèle que lorsque l'on croise les informations de plusieurs tables.

Ils permettent la création de tables temporaires regroupant les informations de plusieurs tables.

Il existe plusieurs façons de regrouper les tables, la plus simple étant un produit cartésien :



Combiner les tables

TABLE Avions :

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

TABLE Compagnies :

| Code | Compagnie | Modèle | Nombre |
|------|------------|----------|--------|
| AF | Air France | 777-200 | 25 |
| AF | Air France | A321-200 | 15 |
| AF | Air France | A330-200 | 15 |
| AF | Air France | A340-300 | 12 |
| AC | Air Canada | 777-200 | 6 |
| AC | Air Canada | A321-200 | 10 |
| AC | Air Canada | CRJ-100 | 51 |

Produit cartésien de deux tables

TABLE Avions, Compagnies :

| Modèle | Avions | | | | Compagnies | | | |
|----------|------------|---------|--------|-----------|------------|------------|----------|--------|
| | Fabricant | Moteurs | Places | Autonomie | Code | Compagnie | Modèle | Nombre |
| 737-900 | Boeing | 2 | 215 | 5926 | AF | Air France | 777-200 | 25 |
| 777-200 | Boeing | 2 | 440 | 14307 | AF | Air France | 777-200 | 25 |
| A321-200 | Airbus | 2 | 220 | 5926 | AF | Air France | 777-200 | 25 |
| A330-200 | Airbus | 2 | 380 | 13890 | AF | Air France | 777-200 | 25 |
| A340-300 | Airbus | 4 | 440 | 13705 | AF | Air France | 777-200 | 25 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | AF | Air France | 777-200 | 25 |
| 737-900 | Boeing | 2 | 215 | 5926 | AF | Air France | A321-200 | 15 |
| 777-200 | Boeing | 2 | 440 | 14307 | AF | Air France | A321-200 | 15 |
| A321-200 | Airbus | 2 | 220 | 5926 | AF | Air France | A321-200 | 15 |
| A320-200 | Airbus | 2 | 380 | 13890 | AF | Air France | A321-200 | 15 |
| A340-300 | Airbus | 4 | 440 | 13705 | AF | Air France | A321-200 | 15 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | AF | Air France | A321-200 | 15 |
| 737-900 | Boeing | 2 | 215 | 5926 | AF | Air France | A330-200 | 15 |
| 777-200 | Boeing | 2 | 440 | 14307 | AF | Air France | A330-200 | 15 |
| A321-200 | Airbus | 2 | 220 | 5926 | AF | Air France | A330-200 | 15 |
| A330-200 | Airbus | 2 | 380 | 13890 | AF | Air France | A330-200 | 15 |
| A340-300 | Airbus | 4 | 440 | 13705 | AF | Air France | A330-200 | 15 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | AF | Air France | A330-200 | 15 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |

(et 24 autres)

Puisque les tables peuvent avoir des attributs portant le même nom, on fait précéder le nom de l'attribut par le nom de la table pour les distinguer (Avions.Modèle, Compagnie.Code, etc.)

La table obtenue par le produit cartésien de deux tables n'a fréquemment guère de sens.

En effet, Si un même attribut est présent dans les deux tables (ici Avions.Modèle et Compagnie.Modèle), il est plus logique de ne s'intéresser qu'aux lignes où les deux attributs sont les mêmes.

Donner du sens au produit cartésien

SELECT * FROM Avions, Compagnies

WHERE Avions.Modèle = Compagnie.Modèle

| Avions | | | | | Compagnies | | | |
|----------|------------|---------|--------|-----------|------------|------------|----------|--------|
| Modèle | Fabricant | Moteurs | Places | Autonomie | Code | Compagnie | Modèle | Nombre |
| 737-900 | Boeing | 2 | 215 | 5926 | AF | Air France | 777-200 | 25 |
| 777-200 | Boeing | 2 | 440 | 14307 | AF | Air France | 777-200 | 25 |
| A321-200 | Airbus | 2 | 220 | 5926 | AF | Air France | 777-200 | 25 |
| A330-200 | Airbus | 2 | 380 | 13890 | AF | Air France | 777-200 | 25 |
| A340-300 | Airbus | 4 | 440 | 13705 | AF | Air France | 777-200 | 25 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | AF | Air France | 777-200 | 25 |
| 737-900 | Boeing | 2 | 215 | 5926 | AF | Air France | A321-200 | 15 |
| 777-200 | Boeing | 2 | 440 | 14307 | AF | Air France | A321-200 | 15 |
| A321-200 | Airbus | 2 | 220 | 5926 | AF | Air France | A321-200 | 15 |
| A330-200 | Airbus | 2 | 380 | 13890 | AF | Air France | A321-200 | 15 |
| A340-300 | Airbus | 4 | 440 | 13705 | AF | Air France | A321-200 | 15 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | AF | Air France | A321-200 | 15 |
| 737-900 | Boeing | 2 | 215 | 5926 | AF | Air France | A330-200 | 15 |
| 777-200 | Boeing | 2 | 440 | 14307 | AF | Air France | A330-200 | 15 |
| A321-200 | Airbus | 2 | 220 | 5926 | AF | Air France | A330-200 | 15 |
| A330-200 | Airbus | 2 | 380 | 13890 | AF | Air France | A330-200 | 15 |
| A340-300 | Airbus | 4 | 440 | 13705 | AF | Air France | A330-200 | 15 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | AF | Air France | A330-200 | 15 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |

(et 24 autres)

On préférera effectuer une *jointure interne* des tables grâce au mot-clé JOIN (ou INNER JOIN)

```
SELECT * FROM Avions
```

```
JOIN Compagnies ON Avions.Modèle = Compagnies.Modèle
```

| Avions | | | | | Compagnies | | | |
|----------|------------|---------|--------|-----------|------------|------------|----------|--------|
| Modèle | Fabricant | Moteurs | Places | Autonomie | Code | Compagnie | Modèle | Nombre |
| 777-200 | Boeing | 2 | 440 | 14307 | AF | Air France | 777-200 | 25 |
| A321-200 | Airbus | 2 | 220 | 5926 | AF | Air France | A321-200 | 15 |
| A330-200 | Airbus | 2 | 380 | 13890 | AF | Air France | A330-200 | 15 |
| A340-300 | Airbus | 4 | 440 | 13705 | AF | Air France | A340-300 | 12 |
| 777-200 | Boeing | 2 | 440 | 14307 | AC | Air Canada | 777-200 | 6 |
| A321-200 | Airbus | 2 | 220 | 5926 | AC | Air Canada | A321-200 | 10 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | AC | Air Canada | CRJ-100 | 51 |

Une jointure interne donne le même résultat qu'un filtrage sur un produit cartésien, mais peut être plus lisible

On peut joindre plus de deux tables :

```
SELECT * FROM table1 JOIN table2 ON table1.x = table2.y  
        JOIN table3 ON table2.z = table3.t
```

On peut utiliser AS pour définir des alias pour les noms des tables :

```
SELECT * FROM Avions AS a
```

```
JOIN Compagnies AS c ON a.Modèle = c.Modèle
```

| Avions / a | | | | | Compagnies / c | | | |
|------------|------------|---------|--------|-----------|----------------|------------|----------|--------|
| Modèle | Fabricant | Moteurs | Places | Autonomie | Code | Compagnie | Modèle | Nombre |
| 777-200 | Boeing | 2 | 440 | 14307 | AF | Air France | 777-200 | 25 |
| A321-200 | Airbus | 2 | 220 | 5926 | AF | Air France | A321-200 | 15 |
| A330-200 | Airbus | 2 | 380 | 13890 | AF | Air France | A330-200 | 15 |
| A340-300 | Airbus | 4 | 440 | 13705 | AF | Air France | A340-300 | 12 |
| 777-200 | Boeing | 2 | 440 | 14307 | AC | Air Canada | 777-200 | 6 |
| A321-200 | Airbus | 2 | 220 | 5926 | AC | Air Canada | A321-200 | 10 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | AC | Air Canada | CRJ-100 | 51 |

Requis si l'on joint plusieurs fois la même table.

Il existe d'autres jointures, notamment pour les cas où les éléments d'une table n'ont pas de correspondance dans l'autre table.

Le produit cartésien est un type de jointure (CROSS JOIN)

Jointure à gauche

On peut utiliser une jointure à gauche (LEFT JOIN ou LEFT INNER JOIN) pour ajouter à la jointure les éléments de la première table qui n'ont pas été associés à un élément de la seconde

```
SELECT * FROM Avions AS a
```

```
LEFT JOIN Compagnies AS c ON a.Modèle = c.Modèle
```

| Avions / a | | | | | Compagnies / c | | | |
|------------|------------|---------|--------|-----------|----------------|------------|----------|--------|
| Modèle | Fabricant | Moteurs | Places | Autonomie | Code | Compagnie | Modèle | Nombre |
| 777-200 | Boeing | 2 | 440 | 14307 | AF | Air France | 777-200 | 25 |
| A321-200 | Airbus | 2 | 220 | 5926 | AF | Air France | A321-200 | 15 |
| A330-200 | Airbus | 2 | 380 | 13890 | AF | Air France | A330-200 | 15 |
| A340-300 | Airbus | 4 | 440 | 13705 | AF | Air France | A340-300 | 12 |
| 777-200 | Boeing | 2 | 440 | 14307 | AC | Air Canada | 777-200 | 6 |
| A321-200 | Airbus | 2 | 220 | 5926 | AC | Air Canada | A321-200 | 10 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 | AC | Air Canada | CRJ-100 | 51 |
| 737-900 | Boeing | 2 | 215 | 5926 | | | | |

Les cases « manquantes » sont vides (contiennent NULL)

Combiner les requêtes

SELECT * FROM Avions WHERE Places > 250

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

SELECT * FROM Avions WHERE Fabricant = Airbus

| Modèle | Fabricant | Moteurs | Places | Autonomie |
|----------|------------|---------|--------|-----------|
| 737-900 | Boeing | 2 | 215 | 5926 |
| 777-200 | Boeing | 2 | 440 | 14307 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| CRJ-100 | Bombardier | 2 | 50 | 1574 |

Union de requêtes $(Req_1 \cup Req_2)$

| | | | | |
|----------|--------|---|-----|-------|
| 777-200 | Boeing | 2 | 440 | 14307 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |

SELECT * FROM Avions WHERE Places > 250

UNION

SELECT * FROM Avions WHERE Fabricant = Airbus

| | | | | |
|----------|--------|---|-----|-------|
| 777-200 | Boeing | 2 | 440 | 14307 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| A321-200 | Airbus | 2 | 220 | 5926 |

Il doit y avoir le même nombre de colonnes, et les colonnes doivent être de même type, mais pas nécessairement contenir les mêmes attributs.

Les lignes apparaissant dans les deux requêtes ne sont retournées qu'une seule fois (utiliser UNION ALL sinon).

Intersection de requêtes $(Req_1 \cap Req_2)$

| | | | | |
|----------|--------|---|-----|-------|
| 777-200 | Boeing | 2 | 440 | 14307 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |

SELECT * FROM Avions WHERE Places > 250

INTERSECT

SELECT * FROM Avions WHERE Fabricant = Airbus

| | | | | |
|----------|--------|---|-----|-------|
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |

Différence de requêtes (Req₁ – Req₂)

| | | | | |
|----------|--------|---|-----|-------|
| 777-200 | Boeing | 2 | 440 | 14307 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |
| A321-200 | Airbus | 2 | 220 | 5926 |
| A330-200 | Airbus | 2 | 380 | 13890 |
| A340-300 | Airbus | 4 | 440 | 13705 |

SELECT * FROM Avions WHERE Places > 250

MINUS (ou EXCEPT)

SELECT * FROM Avions WHERE Fabricant = Airbus

| | | | | |
|---------|--------|---|-----|-------|
| 777-200 | Boeing | 2 | 440 | 14307 |
|---------|--------|---|-----|-------|

Création d'une table

Le mot-clé CREATE permet de créer une table.

On spécifie la liste des attributs et leur domaine

(INTEGER pour un entier, VARCHAR pour une chaîne de caractères, FLOAT pour un réel, BOOLEAN pour une valeur booléenne...)

```
CREATE TABLE IF NOT EXISTS Avions (  
    Modèle      VARCHAR(6) PRIMARY KEY,  
    Fabricant   VARCHAR(12),  
    Moteurs     INTEGER,  
    Places     INTEGER,  
    Autonomie  FLOAT );
```

Taille et précision des attributs

Certains types ont des arguments pour préciser la longueur maximale, par exemple les chaînes de caractères.

VARCHAR(6) correspond à des chaînes d'au plus 6 caractères.

Pour les entiers, le chiffre n'indique que le nombre de caractères lors de l'affichage, mais pas les valeurs admises.

INTEGER(2) peut contenir des entiers de plus de deux chiffres (selon le SGBD), mais certains *affichages* seront réduits à deux chiffres.

DECIMAL(5,2) ou NUMERIC(5,2) désigne un nombre à virgule avec au plus 5 chiffres, dont deux après la virgule (donc de -999.99 à +999.99).

Il est fréquent que l'on ait besoin de pouvoir faire référence à un ligne dans une table *de manière unique*.

On peut définir une *clé primaire* qui est un attribut (ou un groupe d'attributs) qui ne peut (peuvent) pas prendre plusieurs fois la (les) même(s) valeur(s) dans la table. Cette unicité est garantie par le SGBD.

Pour une clé primaire constituée d'un unique attribut :

```
Modèle VARCHAR(6) PRIMARY KEY,
```

Pour une clé primaire consistant en un groupe d'attributs :

```
Modèle VARCHAR(6),
```

```
Fabricant VARCHAR(12),
```

```
PRIMARY KEY(Modèle, Fabricant)
```

Les autres attributs d'une table sont *fonctionnellement dépendants* de la clé primaire

Avoir une clé primaire dans un GROUP BY autorise à projeter sur tous les attributs de la table (SQL récent et SGBD compatibles)

Il y a *au plus* une clé primaire (*primary key*) par table.

Si aucun attribut ou groupe d'attributs n'est satisfaisant pour cet usage, on utilise souvent un entier à cet effet (sans sens particulier, mais unique pour chaque enregistrement).

Un enregistrement ne peut avoir « NULL » pour un attribut faisant partie d'une clé primaire.

Il n'y a pas de contrainte particulière sur les valeurs des autres attributs d'un enregistrement, excepté :

- on peut empêcher que les enregistrements puisse ne pas avoir de valeur pour un attribut donné en indiquant NOT NULL dans la définition de l'attribut ;
- on peut limiter les valeurs possibles d'un attribut aux valeurs présentes dans une autre table en définissant une relation de *clé étrangère*.

Il est fréquent que certains attributs apparaissent dans plusieurs tables pour permettre des jointures (pas nécessairement sous le même nom !)

Pour *diminuer les risques de données incorrectes*, on peut déclarer ces relations, en spécifiant qu'un attribut d'une table fait référence à un attribut (souvent une clé primaire) d'une autre table.

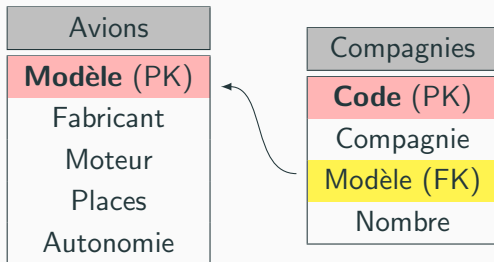
C'est ce que l'on appelle *clé étrangère* (*foreign key*), définie par :

```
FOREIGN KEY (attribut) REFERENCES autre_table(attribut)
```

Seules les valeurs effectivement prises par l'attribut dans l'autre table seront acceptées pour l'attribut concerné (ou NULL).

Par exemple, pour la table Compagnie :

```
CREATE TABLE IF NOT EXISTS Compagnie (  
    Code          VARCHAR(6) NOT NULL PRIMARY KEY,  
    Compagnie     VARCHAR(18),  
    Modèle       VARCHAR(6),  
    Nombre        INTEGER,  
    FOREIGN KEY  (Modèle) REFERENCES Avion(Modèle) );
```



Les clés étrangères suggèrent une jointure possible :

```
CREATE TABLE IF NOT EXISTS Compagnie (
```

...

```
FOREIGN KEY (Modèle) REFERENCES Avion(Modèle) );
```

```
FROM Compagnie JOIN Avion ON Compagnie.Modèle = Avion.Modèle
```



On peut ajouter des attributs :

```
ALTER TABLE nom_de_la_table  
    ADD nom_nouvel_attribut type_nouvel_attribut
```

Ou bien en supprimer :

```
ALTER TABLE nom_de_la_table  
    DROP nom_attribut
```

Le renommage est possible, mais la syntaxe dépend du SGBD.

Pour ajouter une série de lignes :

```
INSERT INTO nom_de_la_table (attribut1, attribut2, attribut3)
```

```
VALUES
```

```
    (valeur1-A, valeur2-A, valeur3-A),
```

```
    (valeur1-B, valeur2-B, valeur3-B),
```

```
    (valeur1-C, valeur2-C, valeur3-C);
```

Si l'on ne précise pas de liste d'attributs, il faut tous les préciser, dans l'ordre.

Pour modifier une série de lignes :

```
UPDATE nom_de_la_table
```

```
    SET attribut = nouvelle_valeur
```

```
    WHERE expression_booléenne
```

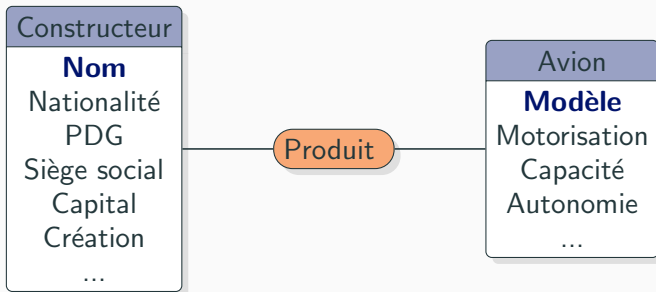
La condition WHERE ... permet de choisir quelles lignes seront mis à jour ou supprimés.

Sans condition, toutes les lignes sont mis à jour/supprimées !

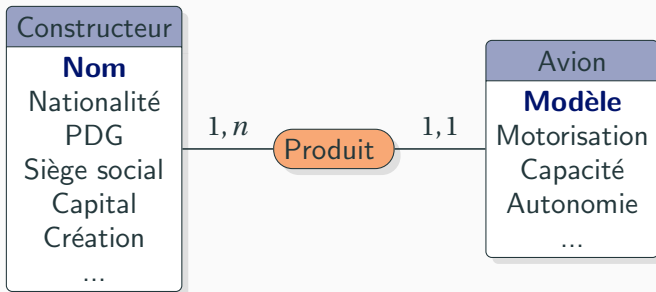
Pour supprimer une série de lignes :

```
DELETE FROM nom_de_la_table  
WHERE expression_booléenne
```

Les « type-entités » sont liés par des « type-associations » :



On peut préciser les cardinalités :



Les possibilités sont :

- 1,1 : en liaison avec une et une seule autre entité
- 1, n : en liaison avec au moins une autre entité
- 0,1 : en liaison avec au plus une autre entité
- 0, n : en liaison avec un nombre quelconque d'autres entités, possiblement aucune

Les type-entités sont convertis en relations (tables)

Lorsque le type-association exhibe une cardinalité 1,1, il est traduit par une clé étrangère :

Constructeur(Nom, Nationalité, PDG, Siège social, Capital, Création, ...)

Avion(Modèle, Motorisation, Capacité, Autonomie, ..., **Constructeur**)

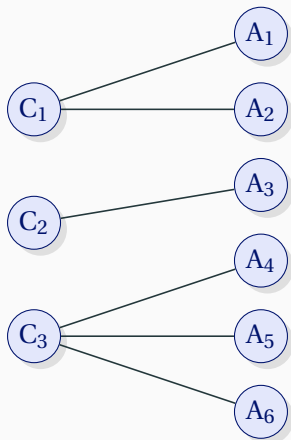
La clé étrangère peut faire partie d'un groupe de clés primaires :

Constructeur(Nom, Nationalité, PDG, Siège social, Capital, Création, ...)

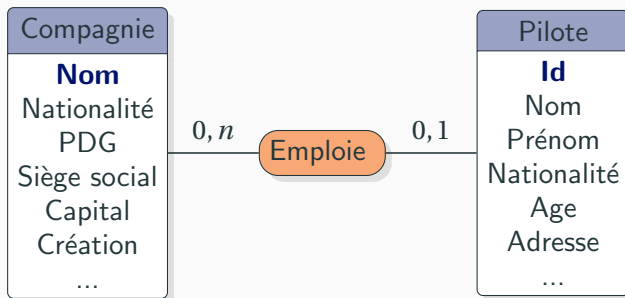
Avion(Modèle, Motorisation, Capacité, Autonomie, ..., **Constructeur**)

Association « one-to-many » (1-*)

Dans le cas des constructeurs et modèles d'appareils :



Un autre exemple

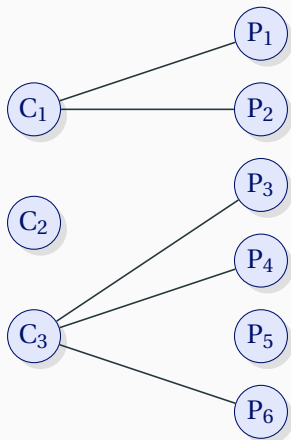


Compagnie(Nom, Nationalité, PDG, Siège social, Capital, Création, ...)

Pilote(Id, Nom, Prénom, Nationalité, Age, Adresse, ..., **Compagnie**)

Un autre exemple

Dans le cas des compagnies et pilotes :



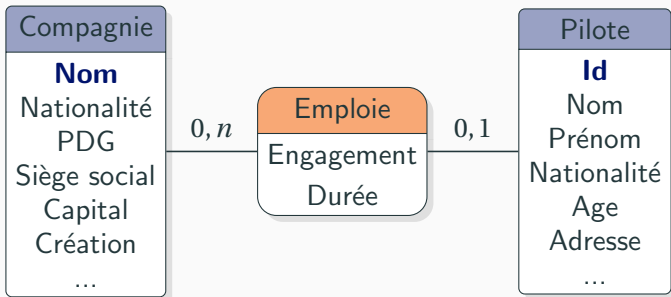
Un autre exemple

On différencie 0,1 et 1,1 avec la clé étrangère :

- dans le cas 0,1, la valeur **NULL** est admise
- dans le cas 1,1, elle ne l'est pas

Il n'est pas facile de signifier la différence entre 0,*n* et 1,*n*

Données dans le type-association



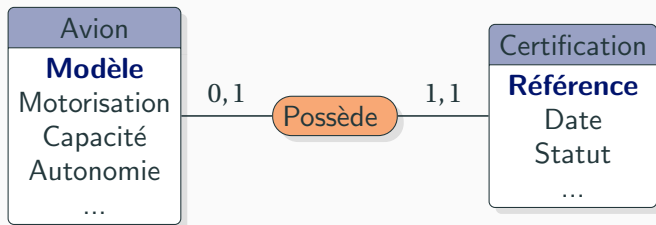
Intégration dans la relation associée au type-entité côté 0,1 ou 1,1 :

Compagnie(Nom, Nationalité, PDG, Siège social, Capital, Création, ...)

Pilote(Id, Nom, Prénom, ..., **Compagnie**, Engagement, Durée)

Association « one-to-one » (1-1)

Par exemple :



Association « one-to-one » (1-1)

S'il y a une cardinalité 1,1 :

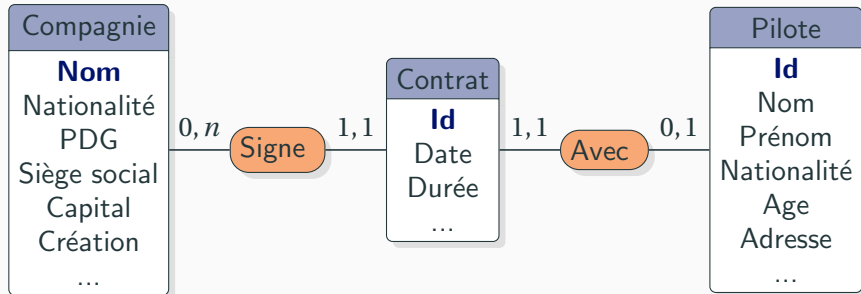
- la clé étrangère peut faire office, seule, de clé primaire
- les données peuvent être intégrées à la relation de l'autre type-entité

Il est difficile de garantir le « au plus un »

Si les deux cardinalités sont 1,1, généralement, on fusionnera les deux type-entités

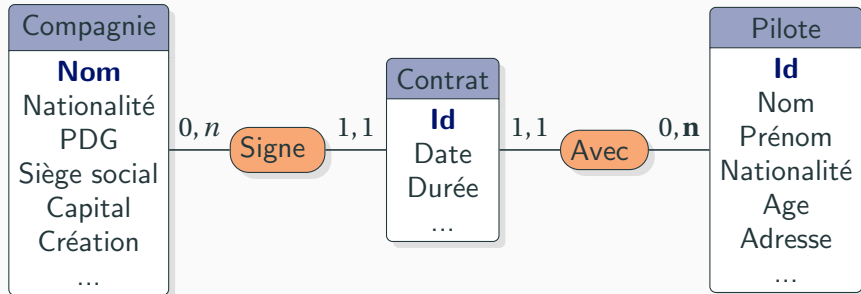
Autre possibilité pour les engagements

On peut vouloir dissocier contrat et pilote :



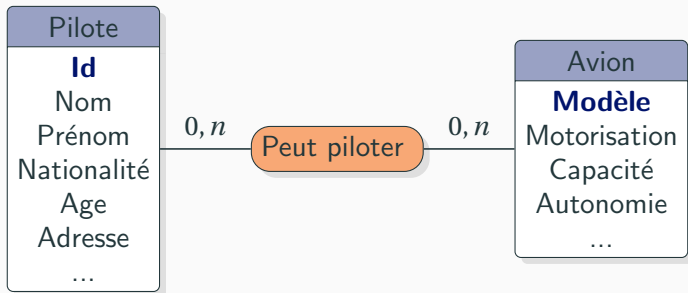
Autre possibilité pour les engagements

Si l'on veut garder trace des contrats passés :



Association « many-to-many » (*-*)

Par exemple :



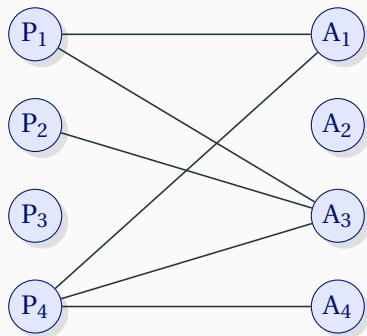
Cette fois, le type-association conduit à une relation

Elle ne contient que deux clés étrangères

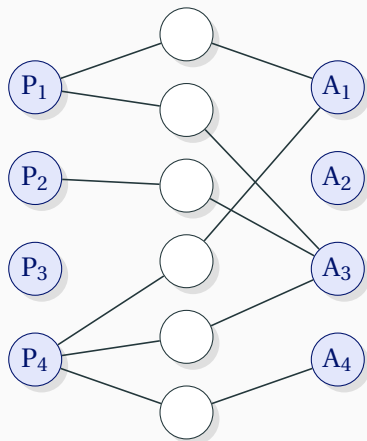
Pilote(Id, Nationalité, Nom, Prénom, Age, ...)

Avion(Modèle, Motorisation, Capacité, Autonomie, ...)

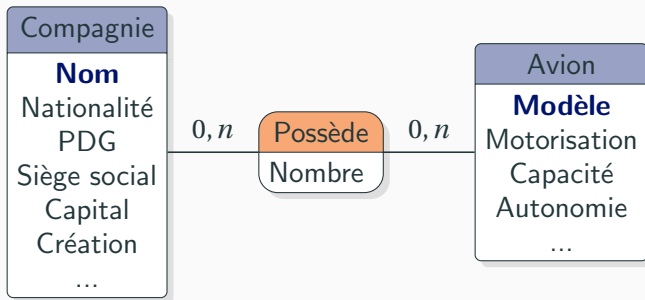
CertificationPilotage(Pilote, Avion)



Décomposition de relations $*_*$



On peut aller un peu plus loin :



Flotte(Compagnie, Avion, Nombre)

Comment ajouter des informations sur les vols ?

- une heure et un aéroport de départ
- une heure et un aéroport d'arrivée
- un modèle d'appareil utilisé
- une compagnie affrétant le vol
- un pilote et un ou plusieurs copilotes