

- **L3 MIAHS/Ingémath**
- **Université Paris Cité**
- Année 2023-2024
- [Course Homepage](#)
- [Moodle](#)



- Pas de documents autorisés
- Pas de téléphone portable



Les trois exercices (modélisation, normalisation, requêtes) portent sur le schéma **nycflights** légèrement nettoyé.

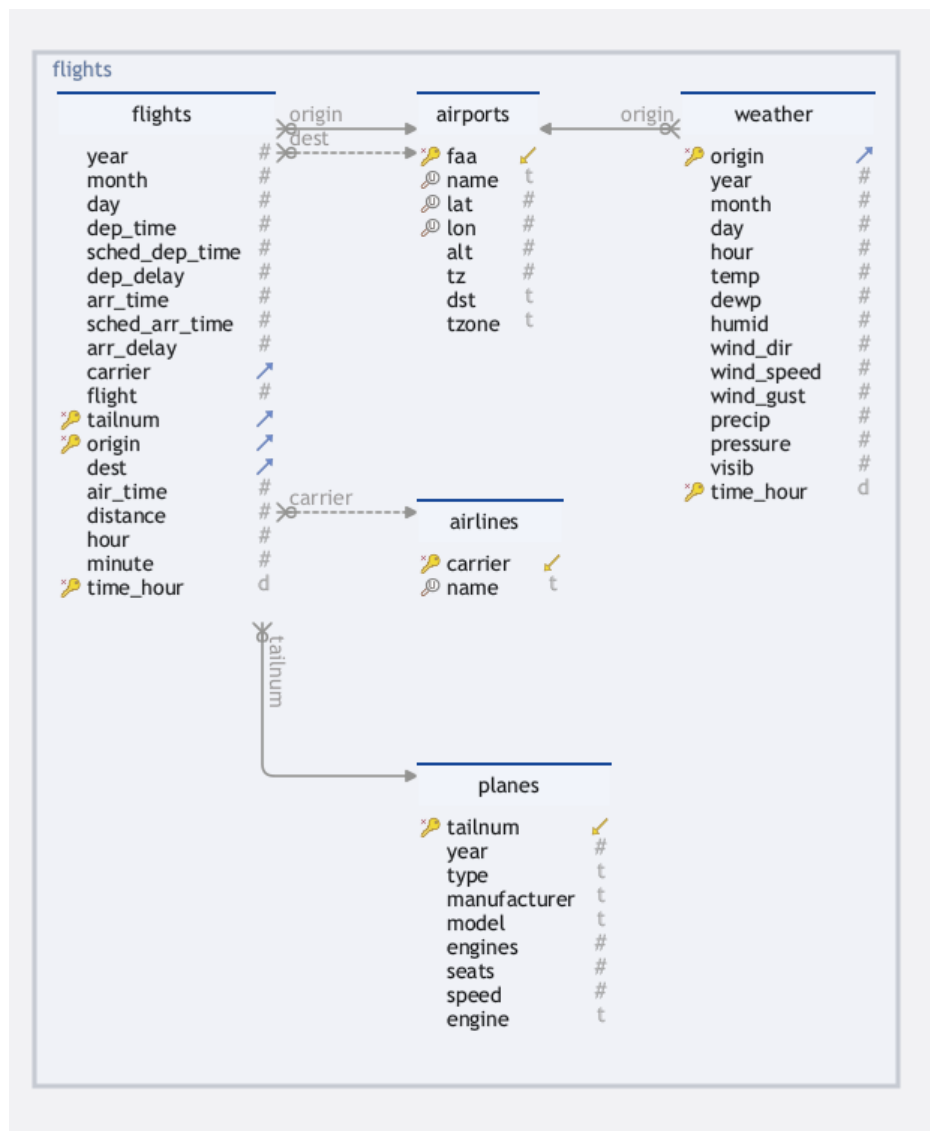


FIG. 1 : NYCFlights en relationnel à pattes de corbeau

## Définition du schéma en SQL

```
CREATE TABLE airlines (  
  carrier text NOT NULL,  
  "name" text NULL,  
  CONSTRAINT airlines_pk  
    PRIMARY KEY (carrier),  
  CONSTRAINT airlines_un  
    UNIQUE (name)  
);
```

```
CREATE TABLE airports (  
  faa text NOT NULL,  
  "name" text NULL,  
  lat float8 NULL,  
  lon float8 NULL,  
  alt float8 NULL,  
  tz float8 NULL,  
  dst text NULL,  
  tzone text NULL,  
  CONSTRAINT airports_pk  
    PRIMARY KEY (faa),  
  CONSTRAINT airports_un  
    UNIQUE (name),  
  CONSTRAINT airports_un_ll  
    UNIQUE (lat, lon)  
);
```

```
CREATE TABLE weather (  
  origin text NOT NULL,  
  "year" int4 NULL,  
  "month" int4 NULL,  
  "day" int4 NULL,  
  "hour" int4 NULL,  
  "temp" float8 NULL,  
  dewp float8 NULL,  
  humid float8 NULL,  
  wind_dir float8 NULL,  
  wind_speed float8 NULL,  
  wind_gust float8 NULL,  
  precip float8 NULL,  
  pressure float8 NULL,  
  visib float8 NULL,  
  time_hour timestamptz NOT NULL,  
  CONSTRAINT weather_pk  
    PRIMARY KEY (origin, time_hour)  
);
```

```
ALTER TABLE weather ADD  
  CONSTRAINT weather_fk  
  FOREIGN KEY (origin)  
  REFERENCES airports(faa)  
  ON DELETE CASCADE  
  ON UPDATE CASCADE;
```

```
CREATE TABLE planes (  
  tailnum text NOT NULL,  
  "year" int4 NULL,  
  "type" text NULL,  
  manufacturer text NULL,  
  model text NULL,  
  engines int4 NULL,  
  seats int4 NULL,  
  speed int4 NULL,  
  engine text NULL,  
  CONSTRAINT planes_pk PRIMARY KEY (tailnum)  
);
```

Dans le schéma nycflights, on a aussi les dépendances fonctionnelles suivantes :

Table airports

- faa, name, et (lon, lat) sont des clés.

Table airlines

- carrier et name sont des clés

Table weather

- origin, time\_hour est une clé
- time\_hour → year, month, day, hour
- year, month, day, hour → time\_hour

Table planes

- tailnum est une clé
- model → manufacturer, engines, engine, type

Table flights

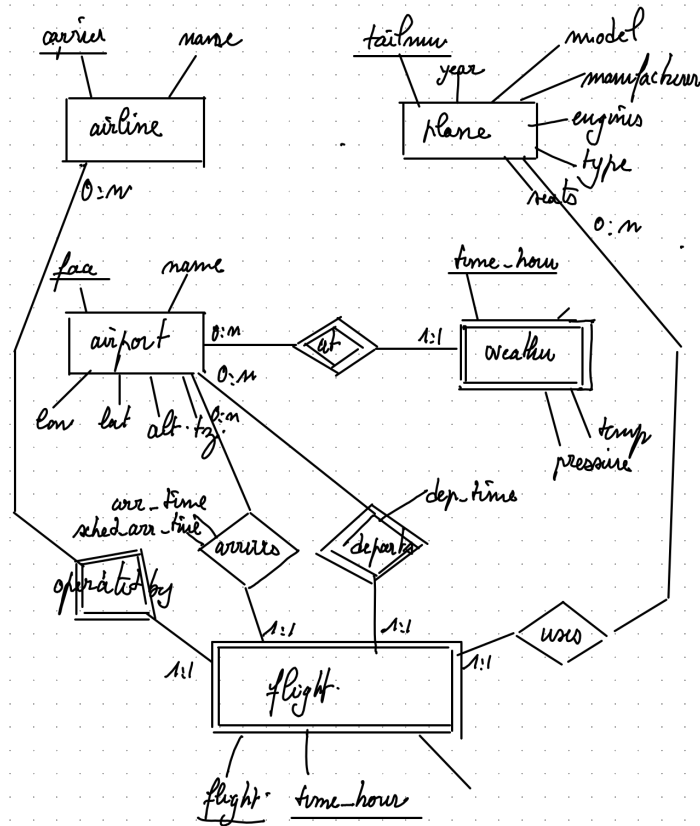
- tailnum, time\_hour → carrier

```
CREATE TABLE flights (  
    "year" int4 NULL,  
    "month" int4 NULL,  
    "day" int4 NULL,  
    dep_time int4 NULL,  
    sched_dep_time int4 NULL,  
    dep_delay float8 NULL,  
    arr_time int4 NULL,  
    sched_arr_time int4 NULL,  
    arr_delay float8 NULL,  
    carrier text NULL,  
    flight int4 NULL,  
    tailnum text NOT NULL,  
    origin text NOT NULL,  
    dest text NULL,  
    air_time float8 NULL,  
    distance float8 NULL,  
    "hour" float8 NULL,  
    "minute" float8 NULL,  
    time_hour timestamptz NOT NULL,  
    CONSTRAINT flights_pk  
        PRIMARY KEY (  
            tailnum, origin, time_hour  
        )  
);  
  
ALTER TABLE flights ADD  
    CONSTRAINT flights_fk  
    FOREIGN KEY (carrier)  
    REFERENCES airlines(carrier)  
    ON DELETE SET NULL  
    ON UPDATE CASCADE;  
  
ALTER TABLE flights ADD  
    CONSTRAINT flights_fk_dest  
    FOREIGN KEY (dest)  
    REFERENCES airports(faa)  
    ON DELETE SET NULL  
    ON UPDATE CASCADE;  
  
ALTER TABLE flights ADD  
    CONSTRAINT flights_fk_origin  
    FOREIGN KEY (origin)  
    REFERENCES airports(faa)  
    ON DELETE SET NULL  
    ON UPDATE CASCADE;  
  
ALTER TABLE flights ADD  
    CONSTRAINT flights_fk_planes  
    FOREIGN KEY (tailnum)  
    REFERENCES planes(tailnum)  
    ON DELETE SET NULL  
    ON UPDATE CASCADE;
```

- time\_hour → sched\_dep\_time
- sched\_dep\_time, dep\_time → dep\_delay
- sched\_arr\_time, arr\_time → arr\_delay
- origin, dest, dep\_time, arr\_time → airtime
- time\_hour → year, month, day, hour, minute
- year, month, day, hour, minute → time\_hour
- origin, dest → distance
- (tailnum, origin, time\_hour) est une clé
- (flight, dest, origin, year, month, day) est une clé

### Exercice : Modélisation

Le schéma entité-association est une tentative de rétro-ingénierie du schéma relationnel nycflights.



#### i Question

Proposez une variation de la représentation de l'entité **plane** où on définit une entité **model** (dont les instances seraient par exemple Airbus A350, Boeing 777, ...), et où chaque avion/aéronef serait considéré comme une réalisation d'une instance de **model**.

Préciser la nature de l'association entre **plane** et **model** et les cardinalités.

Précisez la répartition des attributs entre **plane** et **model**.

### Exercice : Normalisation

#### i Question 1

Pour chaque table, préciser si elle est en FNBC ou non.

#### i Question 2

Si certaines tables ne sont pas en FNBC, proposer une décomposition en FNBC sans perte d'information.

**Exercice : Requêtes (schéma nycflights)**

**i** **Requête 1**

For each airport of departure (denoted by **origin**), for each day of the year, list the codes (**carrier**) of the airlines that have one or more planes taking off from that airport on that day.

**i** **Requête 2**

Lister pour chaque aéroport d'origine, chaque jour de l'année, pour chaque compagnie aérienne, le nombre d'avions exploités par la compagnie aérienne qui décollent de cet aéroport, ce jour là.

**i** **Requête 3**

Lister pour chaque vol exploité par la compagnie (**carrier** nommé dans **airlines**) *Delta Air Lines Inc.* : les conditions météorologiques (**weather**) à l'heure prévue du décollage (**sched\_dep\_time**).

**i** **Requête 4**

Nombre de vols au décollage par aéroport d'origine et par compagnie aérienne (**carrier**).

**i** **Requête 5**

Lister les caractéristiques des avions (**planes**) exploités par au moins deux compagnies aériennes (**carrier**) différentes dans la base de données.

**i** **Requête 6**

Lister pour chaque jour et chaque aéroport d'origine les dix avions les plus en retard au décollage (**dep\_delay**). Ne pas prendre en compte les vols annulés (**dep\_time IS NULL**).

**i** **Requête 7**

Lister pour chaque modèle d'avion (**model**) le nombre de jours où un avion de ce modèle a subi le plus grand retard au décollage (**dep\_delay**) parmi les avions qui ont décollé ce jour là du même aéroport (**origin**).

**i** **Requête 8**

Lister les aéroports de destination (**dest**) qui sont desservis au moins une fois à partir de chaque aéroport de départ (**origin**).

**i** **Requête 9**

Lister les compagnies aériennes (**carrier**) pour lesquelles, chaque jour, au moins un avion figure parmi les 10 avions les plus en retard au décollage (**dep\_delay**) de son aéroport de départ (**origin**).

### Requête 10

Pour chaque couple (`origin`, `dest`), lister les dix vols les plus rapides (`airtime` donne le temps de vol, `distance` la distance entre `dest` et `origin`).

### Quelques conseils

- Préférez les clauses `WITH` et les jointures aux requêtes imbriquées sauf si la requête imbriquée est très simple. C'est une question de lisibilité et donc souvent de correction.
- Ne mélangez pas les fonctions fenêtres et les clauses `GROUP BY ...`.

```
SELECT ..., FOO() OVER w
FROM R
WINDOW w AS (PARTITION BY ... ORDER BY ...)
GROUP BY ... ;
```

est tout simplement incorrect.

- Lorsque vous effectuez un partitionnement par `GROUP BY ...`, la clause `SELECT ...` est sévèrement contrainte, vous n'y trouverez que
  - les colonnes qui ont servi dans la clause `GROUP BY ...`, normalement elles devraient toutes y figurer
  - des fonctions d'aggrégation, comme `COUNT(...)`, `SUM(...)`, `VAR(...)`