Mémo Postgresql

```
q -> quit
\d nom -> affiche table
\g ou ; execute
```

```
psql -U florian -h 127.0.0.1
\i .sql
```

```
CREATE DATABASE xxx WITH OWNER username;
```

=	Equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
<>	Not equal to (ou !=)
LIKE	

```
SELECT first, last, city FROM empinfo WHERE first LIKE 'Er%';
'%s'        '%illia%'
```

```
SELECT * FROM albums WHERE genre IN ('pop','soul');
```

COUNT() - returns the number of rows
SUM() - returns the total sum of a numeric column
AVG() - returns the average of a set of values
MIN() / MAX() - gets the min/max value from a column

```
SELECT artist, album, released FROM albums
WHERE released = (SELECT MIN(released) FROM albums);
```

```
FROM video_games AS games
```

```
UPDATE tv series
SET genre = 'drama'
WHERE id = 2;
```

```
DELETE FROM tv_series
WHERE id = 4
```

```
TRUNCATE/DROP TABLE table_name;
```

```
WHERE ... NOT IN (...)
```

(INNER) JOIN:	Matching in both
LEFT (OUTER) JOIN:	Left table + matched from the right table
RIGHT (OUTER) JOIN:	Right table + matched from the left table
FULL (OUTER) JOIN:	Either left or right table

```
m1.pays < m2.pays
count(...)
```

Plusieurs tables -> GROUP BY

```
SELECT ...
EXCEPT
SELECT ...
```

```
SELECT ... FROM ... WHERE ... GROUP BY ...
HAVING count(*) >= all (SELECT count(*) [...] GROUP BY...)
```

```
CREATE VIEW name AS (SELECT...)
```

```
SELECT DISTINCT
```

```
SELECT groupe, count(*), MAX(ddn), MIN(ddn)
FROM si4 GROUP BY groupe;
```

```
WITH RECURSIVE recursion(liste_champ) AS (
SELECT liste_champ FROM table
UNION ALL
SELECT liste_champ FROM recursion, table WHERE condition)
SELECT * FROM recursion;
```

```
WITH RECURSIVE reaches (departure, escales, arrival) AS
(SELECT departure, 0, arrival FROM vols
UNION
SELECT R1.departure,  1 + R2.escales, R2.arrival
FROM vols AS R1, reaches AS R2
WHERE R1.arrival = R2.departure)
SELECT * FROM reaches where escales > 1;
```

```
WITH RECURSIVE t(n) AS (
SELECT 2
UNION
SELECT n+2 FROM t WHERE n < 100)
SELECT n FROM t;
SELECT count(*), sum(n), max(n), min(n) FROM t;
```

```
DROP VIEW IF EXISTS  ascendant;
CREATE View ascendant(ascendant, enfant) AS (
WITH RECURSIVE ancetre(Aieul, Enfant) AS (
SELECT Pere, Enfant FROM Parents
UNION
SELECT Mere, Enfant FROM Parents
UNION
SELECT A.Aieul, P.Enfant
FROM Parents AS P, ancetre AS A
WHERE P.Pere = A.Enfant OR  P.Mere = A.Enfant )
SELECT * FROM ancetre);
```

```
SELECT * FROM ascendant where Enfant='Julia';
```

```
SELECT * FROM client c
FULL OUTER JOIN telephone t USING (cli_id)
FULL OUTER JOIN email e USING (cli_id)
FULL OUTER JOIN adresse a USING (cli_id)
```