PostgreSQL Notebook

by Rafe

💽 SQL SELECT STATEMENTS

To select all columns in a table:

```
SELECT * FROM table_name;
```

To Select some specific columns:

```
SELECT c1, c2 FROM table_name;
```

To select unique values in column rows:

```
SELECT DISTINCT c1,c2 FROM table_name
```

To select columns c1 and c2 with a some conditions and ordered by an ascending format at column c1 and descending format at column c2:

```
SELECT c1, c2 FROM table_name
WHERE condition1 AND condition2 OR Condition3
ORDER BY c1 ASC, c2 DESC
```

PS: We Also Have `BETWEEN X AND Y` and `LIKE '_XX%'` or `ILIKE '_xx%'` for some extra conditions

---

To **Group By** a table based on a column and do some aggregate function on some other column, one could do:

```
SELECT c1,c3 AGG(c2) FROM table_name
GROUP BY c1,c3
```

- Note that every column that has been selected must be called again after `GROUP BY`
- Note that some aggregate functions are: `SUM()` , `COUNT()` , `AVG()` , `MIN()` , `MAX()` , ...

To filter the aggregated column we use `HAVING` like below:

```
SELECT c1, AGG(c2) FROM table_name
GROUP BY c1
HAVING AGG(c2) >= some_value
```

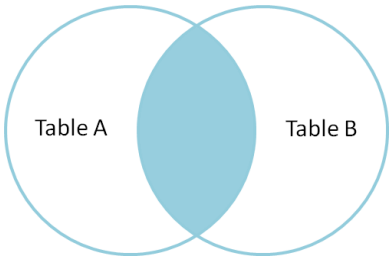
💽 SQL UPDATE TABLE

To use an alias for a column:

```
SELECT c1 AS new_name, AGG(c2) AS new_name2
FROM table_name
GROUP BY c1
HAVING AGG(c2) > some_val
```

- Note that `AS` command work as the very last command so you must use **the initial column name** for filtering on that column.

INNER JOIN



The desired column should be selected to do inner join on it. Other columns could be selected for farther intuition on the data:

```
SELECT payment.customer_id, customer.email, SUM(amount) AS tot_paid
FROM payment
INNER JOIN customer
ON payment.customer_id = customer.customer_id
GROUP BY payment.customer_id, customer.email
HAVING SUM(amount) > 100
```

- Note that in the inner join there is no difference on choosing which table first.

OUTER JOIN