SERIOUS SQL LIVE WEEK 1: 20TH NOV

BY DANNY MA

AGENDA:

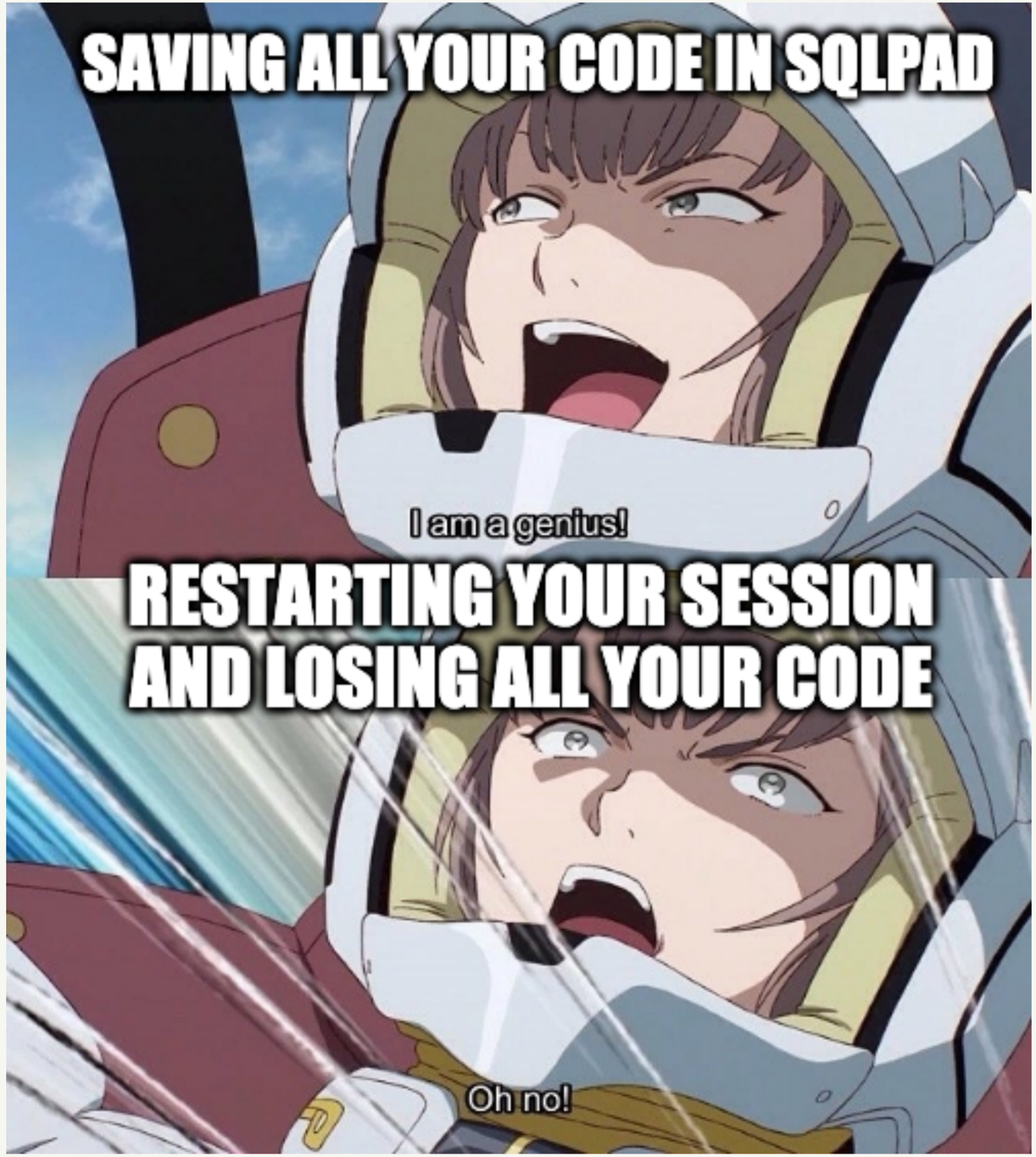
- Intro (10 mins)
- Select & Sort (20 mins)
- Record Counts & Distinct Values (30 mins)

SQL CODING SETUP

- Docker
- Markdown Notes
- VS Code / Sublime / Atom / ???
- SQLPad — localhost:3000

code chunks
notes

DO NOT SAVE YOUR CODE IN SQLPAD

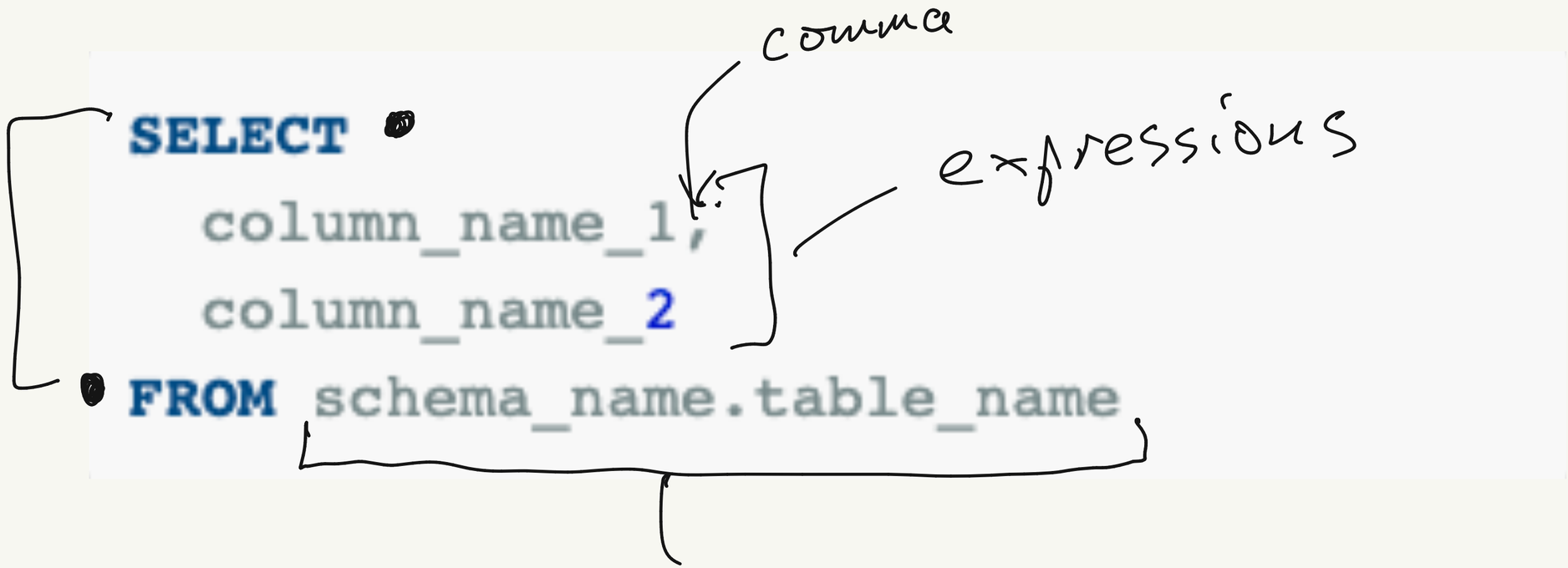


LEARNING MARKDOWN

- Learn by example
- Code + Notes + Images + Links
- Cheatsheets + GitHub Repos
- Try it out with Stack Edit

SELECT & SORT

SELECT STATEMENTS



SELECT * (STAR)

Example Exercise

> Show all records from the `language` table from the `dvd_rentals` schema

```
SELECT * FROM dvd_rentals.language;
```

SELECT COLUMNS

> Show only the `language_id` and `name` columns from the `language` table

```
SELECT
  language_id,
  name
FROM dvd_rentals.language;
```

LIMIT OUTPUT

> Show the first 10 rows from the `actor` tables

```
SELECT *
FROM dvd_rentals.actor
LIMIT 10;
```

top, head

USE LIMIT AS DEFAULT

SORTING TEXT COLUMN

What are the first 5 values in the `country` column from the `country` table by alphabetical order?

```
SELECT country
FROM dvd_rentals.country
ORDER BY country
LIMIT 5;
```

SORT AND LIMIT

> What are the 5 lowest `total_sales` values in the `sales_by_film_category` table?

```
SELECT
  total_sales
FROM dvd_rentals.sales_by_film_category
ORDER BY 1
LIMIT 5;
```

SORT DESCENDING

> What are the first 5 values in **reverse** alphabetical order in the `country` column from the `country` table?

```
SELECT country
FROM dvd_rentals.country
ORDER BY country DESC
LIMIT 5;
```

SORT WITH MULTIPLE COLUMNS

> Which `category` had the lowest `total_sales` value according to the `sales_by_film_category` table? What was the `total_sales` value?

```
SELECT
  category,
  total_sales
FROM dvd_rentals.sales_by_film_category
ORDER BY total_sales
LIMIT 1;
```

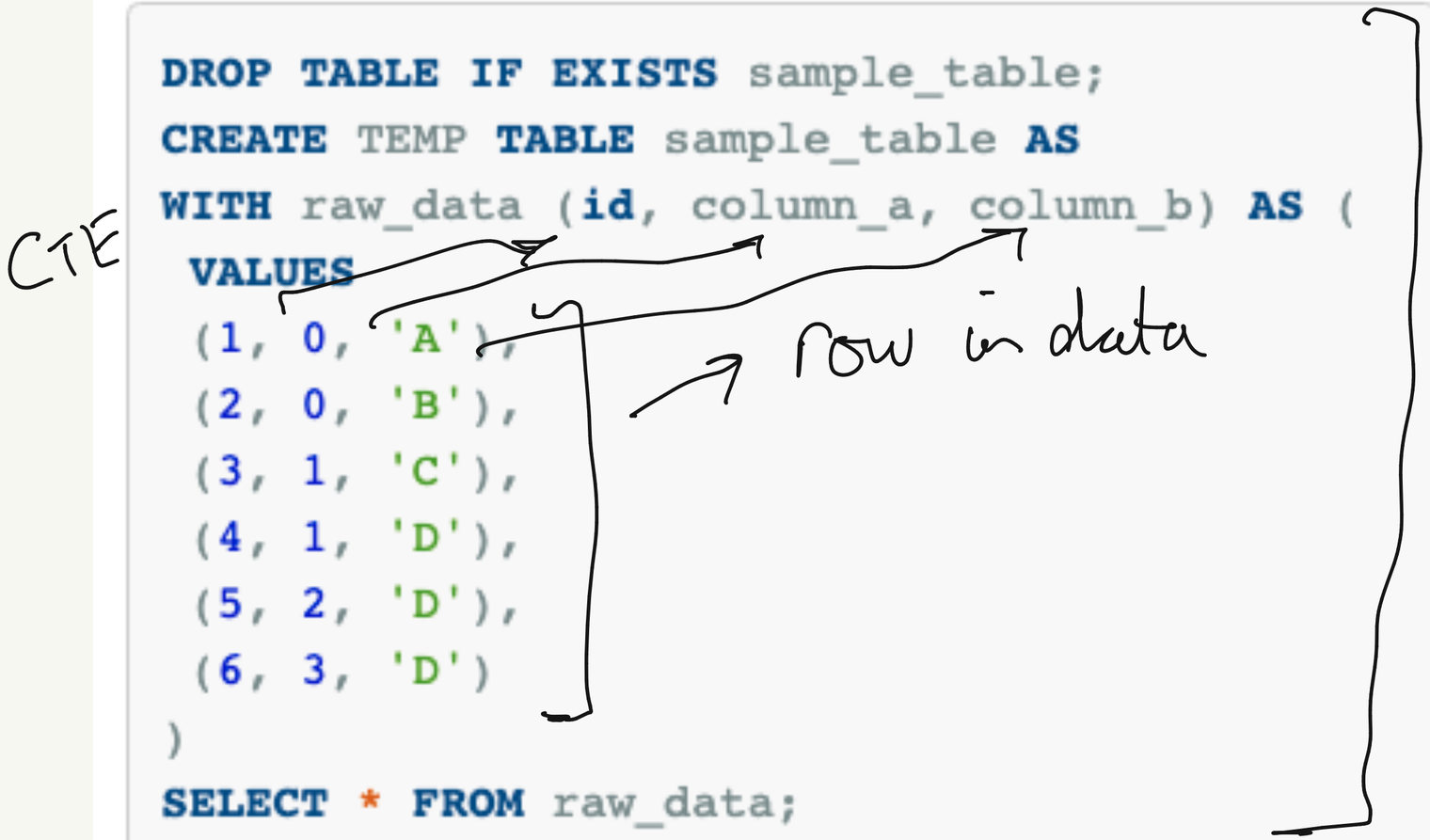
SORT BY DESCENDING LIMIT 1

> What was the latest `payment_date` of all dvd rentals in the `payment` table?

```
SELECT
  payment_date
FROM dvd_rentals.payment
ORDER BY payment_date DESC
LIMIT 1;
```

SORT BY MULTIPLE COLUMNS

```sql
DROP TABLE IF EXISTS sample_table;
CREATE TEMP TABLE sample_table AS
WITH raw_data (id, column_a, column_b) AS (
 VALUES
 (1, 0, 'A'),
 (2, 0, 'B'),
 (3, 1, 'C'),
 (4, 1, 'D'),
 (5, 2, 'D'),
 (6, 3, 'D')
)
SELECT * FROM raw_data;
```



sample_table

id	column_a	column_b
1	0	A
2	0	B
3	1	C
4	1	D
5	2	D
6	3	D

SORT BY 2 COLUMNS ASCENDING

```
SELECT * FROM sample_table
ORDER BY column_a, column_b
```

asc

id	column_a	column_b
1	0	A
2	0	B
3	1	C
4	1	D
5	2	D
6	3	D

SORT ASCENDING & DESCENDING

```
SELECT * FROM sample_table
ORDER BY column_a DESC, column_b
```

id	column_a	column_b
6	3	D
5	2	D
3	1	C
4	1	D
1	0	A
2	0	B

SORT BOTH DESCENDING

```
SELECT * FROM sample_table
ORDER BY column_a DESC, column_b DESC
```

id	column_a	column_b
6	3	D
5	2	D
4	1	D
3	1	C
2	0	B
1	0	A

CHANGED COLUMN ORDER

```
SELECT * FROM sample_table
ORDER BY column_b DESC, column_a;
```

id	column_a	column_b
4	1	D
5	2	D
6	3	D
3	1	C
2	0	B
1	0	A

CHANGED COLUMN ORDER AGAIN

```
SELECT * FROM sample_table
ORDER BY column_b, column_a DESC;
```

id	column_a	column_b
1	0	A
2	0	B
3	1	C
6	3	D
5	2	D
4	1	D

RECORD COUNTS & DISTINCT VALUES

HOW MANY RECORDS?

How many rows are there in the `film_list` table?

count(1)
count(column_name)

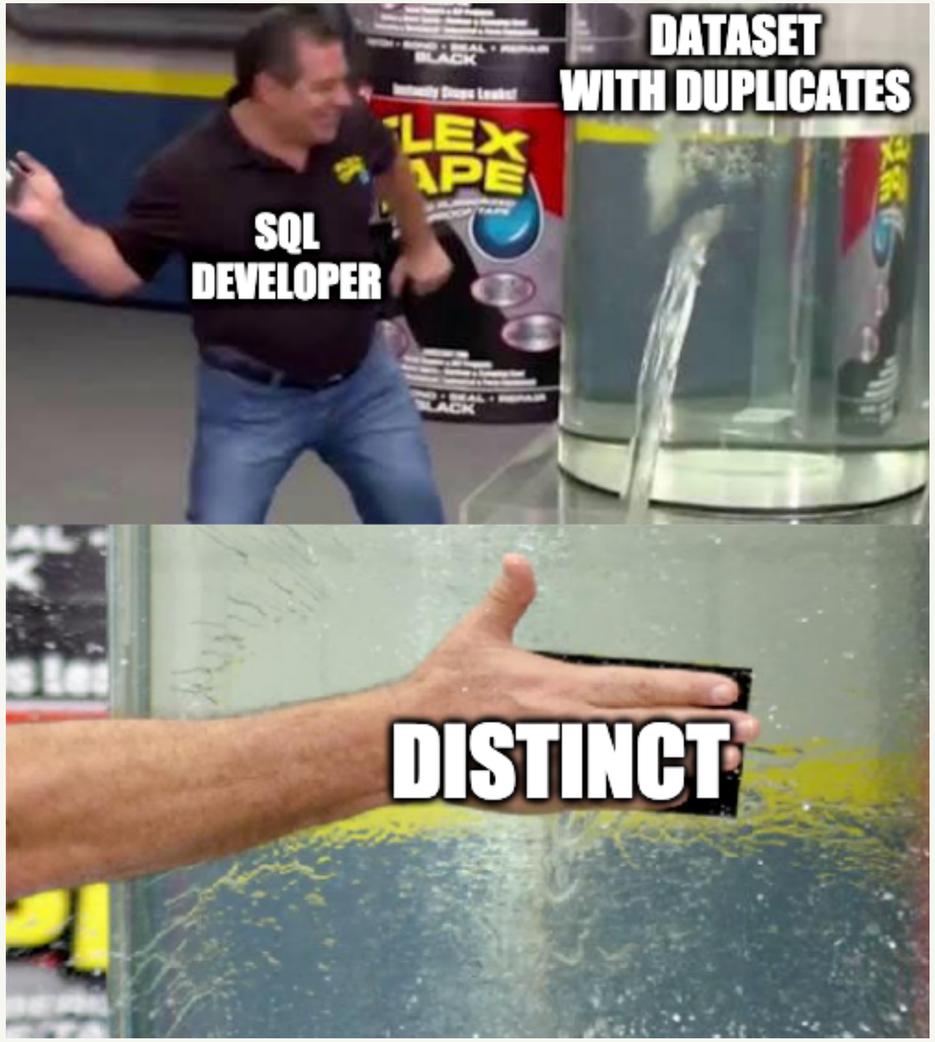
alias

```
SELECT
   COUNT(*) AS row_count
FROM dvd_rentals.film_list;
```

COLUMN ALIASES

```sql
SELECT
   COUNT(*) AS row_count
FROM dvd_rentals.film_list;
```

DISTINCT IN A NUTSHELL



UNIQUE COLUMN VALUES

> What are the unique values for the `rating` column in the `film` table?

```
SELECT DISTINCT :
  rating
FROM dvd_rentals.film_list
```

COUNT OF UNIQUE VALUES

> How many unique `category` values are there in the `film_list` table?

count(*) — row count

```
SELECT
  COUNT(DISTINCT category) AS unique_category_count
FROM dvd_rentals.film_list
```

GROUP BY COUNT

count(*) — not check nulls

> What is the frequency of values in the `rating` column in the `film_list` table?

count(rating)

count(*) <=> count(1)

slower

GROUP BY EXAMPLE SIMPLIFIED TABLE

Simplified `dvd_rentals.film_list` Table Example

fid	title	category	rating	price
730	RIDGEMONT SUBMARINE	New	PG-13	0.99
892	TITANIC BOONDOCK	Animation	R	4.99
286	ENOUGH RAGING	Travel	NC-17	2.99
857	STRICTLY SCARFACE	Comedy	PG-13	2.99
593	MONTEREY LABYRINTH	Horror	G	0.99
664	PATRIOT ROMAN	Action	PG	2.99
211	DARLING BREAKING	Games	PG-13	4.99
932	VALLEY PACKER	Comedy	G	0.99
550	MAGUIRE APACHE	Family	NC-17	2.99
504	KWAI HOMEWARD	Drama	PG-13	0.99

DIVIDING ROWS INTO SUB-GROUPS

PG-13 rows

count(*) => 4

fid	title	category	rating	price
730	RIDGEMONT SUBMARINE	New	PG-13	0.99
857	STRICTLY SCARFACE	Comedy	PG-13	2.99
211	DARLING BREAKING	Games	PG-13	4.99
504	KWAI HOMEWARD	Drama	PG-13	0.99

G rows

count(*) => 2

fid	title	category	rating	price
593	MONTEREY LABYRINTH	Horror	G	0.99
932	VALLEY PACKER	Comedy	G	0.99

GROUP BY RESULTS

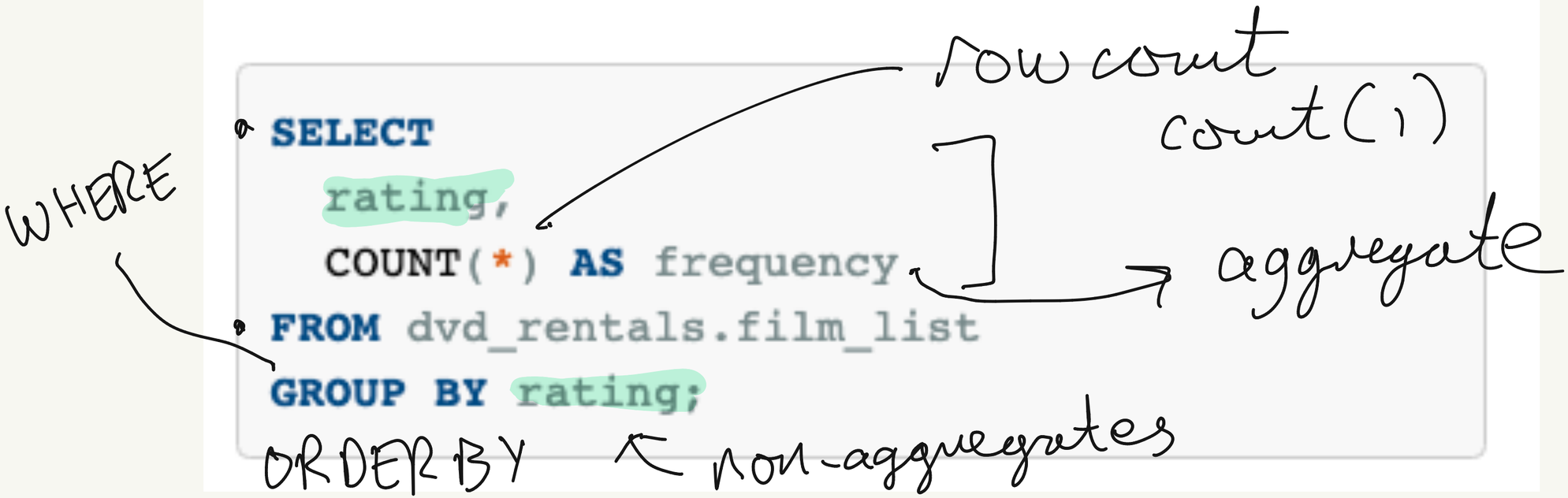
The important thing to note for `GROUP BY` aggregate functions is this:

> Only 1 row is returned for each group 

GROUP BY EXAMPLE

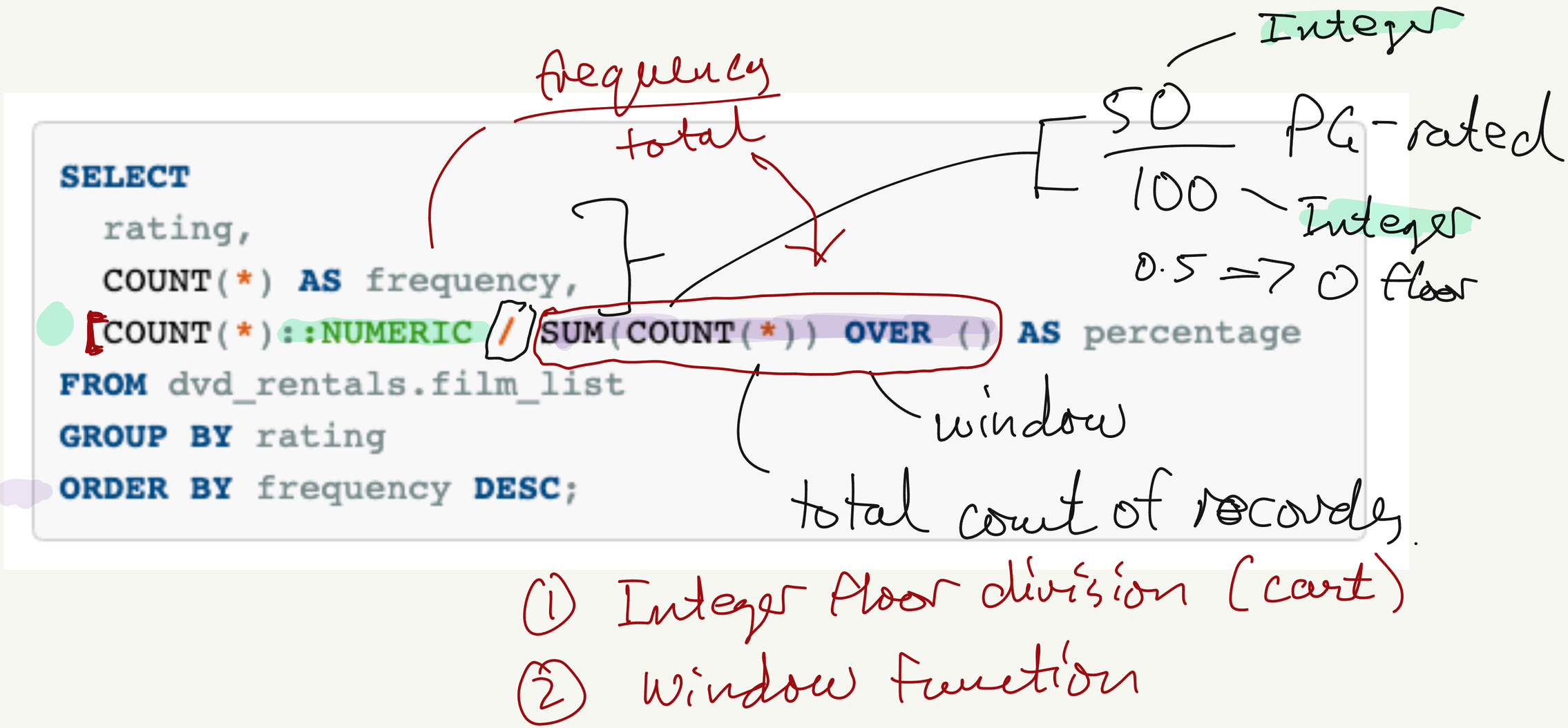
What is the frequency of values in the `rating` column in the `film` table?

```sql
SELECT
    rating,
    COUNT(*) AS frequency
FROM dvd_rentals.film_list
GROUP BY rating;
```



ADDING A PERCENTAGE

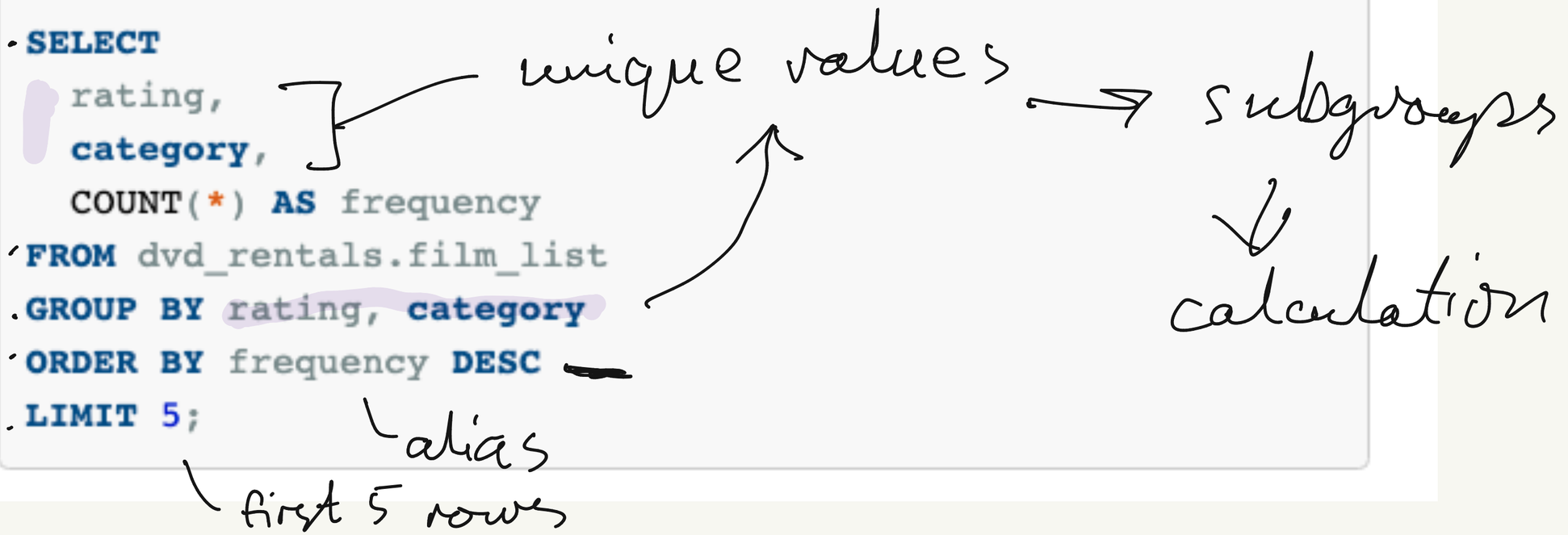
```sql
SELECT
  rating,
  COUNT(*) AS frequency,
  COUNT(*)::NUMERIC / SUM(COUNT(*)) OVER () AS percentage
FROM dvd_rentals.film_list
GROUP BY rating
ORDER BY frequency DESC;
```



MULTIPLE COLUMN GROUP BY

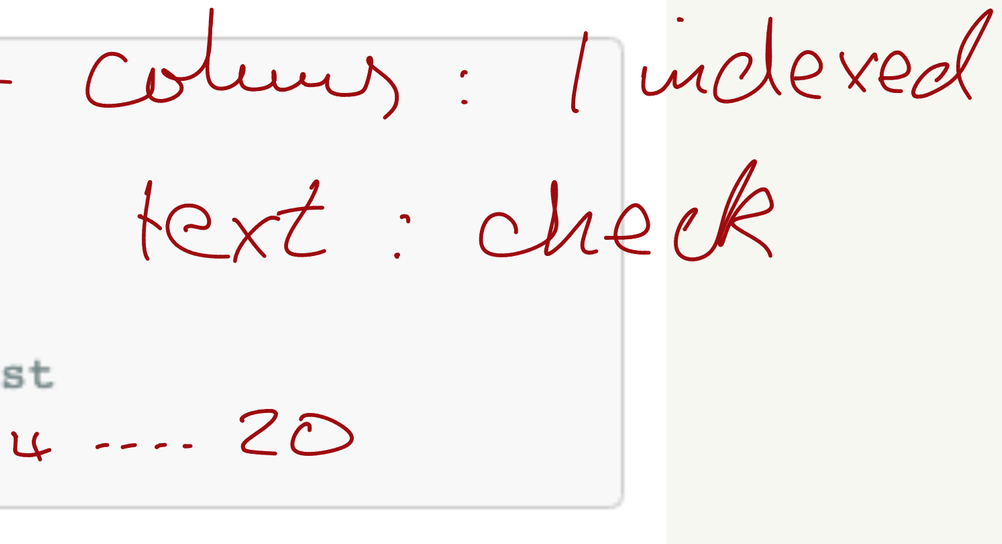
> What are the 5 most frequent `rating` and `category` combinations in the `film_list` table?

```sql
SELECT
    rating,
    category,
    COUNT(*) AS frequency
FROM dvd_rentals.film_list
GROUP BY rating, category
ORDER BY frequency DESC
LIMIT 5;
```



GROUP BY ORDINAL SYNTAX

```
SELECT
  rating,
  category,
  COUNT(*) AS frequency
FROM dvd_rentals.film_list
GROUP BY 1,2
```



GROUP BY SYNTAX (NOT SO GOOD)

```sql
SELECT
  rating,
  category,
  COUNT(*) AS frequency
FROM dvd_rentals.film_list
GROUP BY 1,2
ORDER BY 2,1,3 DESC
LIMIT 5;
```

✓ ✗

category, rating, freq... desc

```sql
SELECT
  rating,
  category,
  COUNT(*) AS frequency
FROM dvd_rentals.film_list
GROUP BY 1,2
ORDER BY 3 DESC
LIMIT 5;
```

frequency DESC

Best practice

RECOMMENDED SYNTAX

```sql
SELECT
  rating,
  category,
  COUNT(*) AS frequency
FROM dvd_rentals.film_list
GROUP BY rating, category
ORDER BY frequency DESC
LIMIT 5;
```

20+