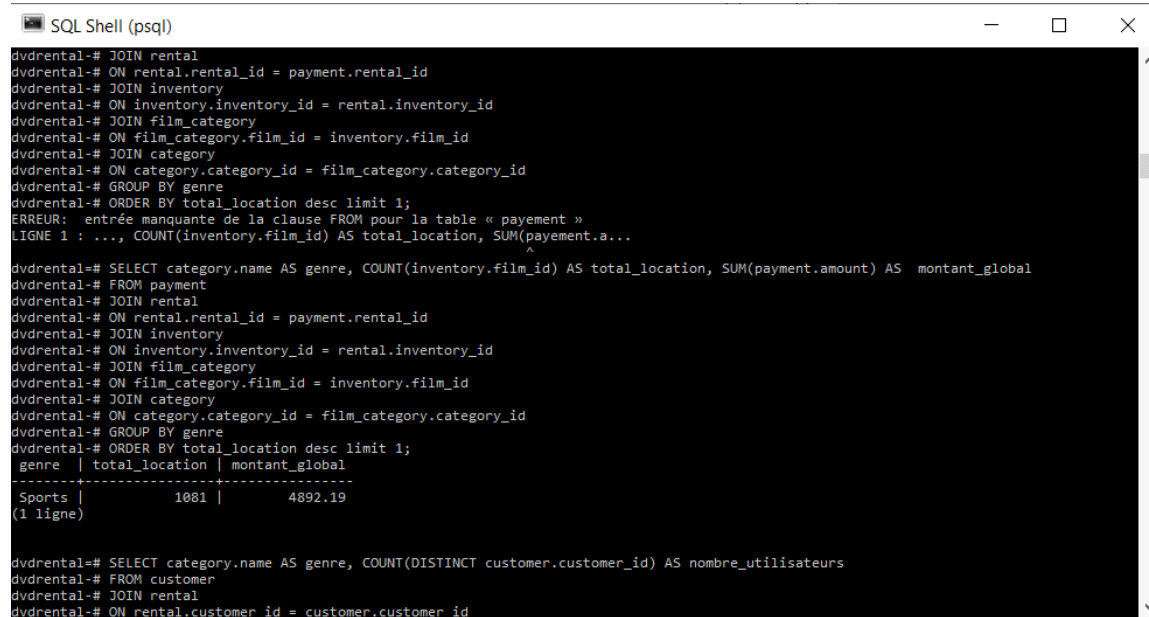


## **BASE DE DONNEE dvdrental:**

1. Quel est la catégorie de film la plus louée, quel est son chiffre d'affaire.

-> SELECT category.name AS genre, COUNT(inventory.film\_id) AS total\_location, SUM(payment.amount) AS montant\_global FROM payment JOIN rental ON rental.rental\_id = payment.rental\_id JOIN inventory ON inventory.inventory\_id = rental.inventory\_id JOIN film\_category ON film\_category.film\_id = inventory.film\_id JOIN category ON category.category\_id = film\_category.category\_id GROUP BY genre ORDER BY total\_location desc limit 1



```
SQL Shell (psql)
dvdrental=# JOIN rental
dvdrental=# ON rental.rental_id = payment.rental_id
dvdrental=# JOIN inventory
dvdrental=# ON inventory.inventory_id = rental.inventory_id
dvdrental=# JOIN film_category
dvdrental=# ON film_category.film_id = inventory.film_id
dvdrental=# JOIN category
dvdrental=# ON category.category_id = film_category.category_id
dvdrental=# GROUP BY genre
dvdrental=# ORDER BY total_location desc limit 1;
ERREUR: entrée manquante de la clause FROM pour la table « payment »
LIGNE 1 : ..., COUNT(inventory.film_id) AS total_location, SUM(payement.a...
^
dvdrental=# SELECT category.name AS genre, COUNT(inventory.film_id) AS total_location, SUM(payment.amount) AS montant_global
dvdrental=# FROM payment
dvdrental=# JOIN rental
dvdrental=# ON rental.rental_id = payment.rental_id
dvdrental=# JOIN inventory
dvdrental=# ON inventory.inventory_id = rental.inventory_id
dvdrental=# JOIN film_category
dvdrental=# ON film_category.film_id = inventory.film_id
dvdrental=# JOIN category
dvdrental=# ON category.category_id = film_category.category_id
dvdrental=# GROUP BY genre
dvdrental=# ORDER BY total_location desc limit 1;
 genre | total_location | montant_global
-----|-----|-----
Sports |          1081 |      4892.19
(1 ligne)
```

2. Combien d'utilisateurs "distincte" ont loué des films d'actions.

-> SELECT category.name AS genre, COUNT(DISTINCT customer.customer\_id) AS nombre\_utilisateurs FROM customer JOIN rental ON rental.customer\_id = customer.customer\_id JOIN inventory ON inventory.inventory\_id = rental.inventory\_id JOIN film\_category ON film\_category.film\_id = inventory.film\_id JOIN category ON category.category\_id = film\_category.category\_id WHERE category.name = 'Action' GROUP BY genre;

```

SQL Shell (psql)
ASTUCE : Peut-être que vous souhaitez référencer la colonne « customer.active ».
dvdrental=# SELECT category.name AS genre, COUNT(DISTINCT customer.customer_id) AS nombre_utilisateurs
dvdrental=# FROM customer
dvdrental=# JOIN rental
dvdrental=# ON rental.customer_id = customer.customer_id
dvdrental=# JOIN inventory
dvdrental=# ON inventory.inventory_id = rental.inventory_id
dvdrental=# JOIN film_category
dvdrental=# ON film_category.film_id = inventory.film_id
dvdrental=# JOIN category
dvdrental=# ON category.category_id = film_category.category_id
dvdrental=# WHERE category.name = 'Action';
ERREUR: la colonne « category.name » doit apparaître dans la clause GROUP BY ou doit être utilisé dans une fonction d'agrégat
LIGNE 1 : SELECT category.name AS genre, COUNT(DISTINCT customer.custo...
dvdrental=# SELECT category.name AS genre, COUNT(DISTINCT customer.customer_id) AS nombre_utilisateurs
dvdrental=# FROM customer
dvdrental=# JOIN rental
dvdrental=# ON rental.customer_id = customer.customer_id
dvdrental=# JOIN inventory
dvdrental=# ON inventory.inventory_id = rental.inventory_id
dvdrental=# JOIN film_category
dvdrental=# ON film_category.film_id = inventory.film_id
dvdrental=# JOIN category
dvdrental=# ON category.category_id = film_category.category_id
dvdrental=# WHERE category.name = 'Action'
dvdrental=# GROUP BY genre;
 genre | nombre_utilisateurs
-----+-----
 Action |          510
(1 ligne)
dvdrental=#

```

3. Déterminer la moyenne de revenu par catégorie ordonnée dans l'ordre décroissant.

-> `SELECT category.name AS genre, ROUND(AVG(payment.amount), 2) AS moyenne FROM payment JOIN rental ON payment.rental_id = rental.rental_id JOIN inventory ON rental.inventory_id = inventory.inventory_id INNER JOIN film_category ON inventory.film_id = film_category.film_id INNER JOIN category ON film_category.category_id = category.category_id GROUP BY genre ORDER BY moyenne;`

```

runpsql - Raccourci
dvdrental=# SELECT category.name AS genre, ROUND(AVG(payment.amount), 2) AS moyenne FROM pay
ment JOIN rental ON payment.rental_id = rental.rental_id JOIN inventory ON rental.inventory_
id = inventory.inventory_id INNER JOIN film_category ON inventory.film_id = film_category.fi
lm_id INNER JOIN category ON film_category.category_id = category.category_id GROUP BY genre
ORDER BY moyenne;
 genre | moyenne
-----+-----
 Children | 3.84
 Family   | 3.88
 Action   | 3.90
 Classics | 3.90
 Animation | 3.99
 Documentary | 4.00
 Music     | 4.10
 Foreign   | 4.13
 Travel    | 4.22
 Drama     | 4.32
 Sci-Fi    | 4.34
 Horror    | 4.40
 Games     | 4.44

```

4. Quels sont les films qui sont retournés en retard.

-> `SELECT film.title, film.rental_duration AS nbre_jour_location_autorise , COUNT(DATE_PART('day',`

rental.return\_date - rental.rental\_date)) AS nbre\_de\_retard FROM rental JOIN inventory ON inventory.inventory\_id = rental.inventory\_id JOIN film ON film.film\_id = inventory.film\_id WHERE DATE\_PART('day', rental.return\_date - rental.rental\_date) > film.rental\_duration GROUP BY film.title, film.rental\_duration ORDER BY title;

runpsql - Raccourci

```
^Cdvdrental=# SELECT film.title, film.rental_duration AS nbre_jour_location_autorise, COUNT(
DATE_PART('day', rental.return_date - rental.rental_date)) AS nbre_de_retard FROM rental JO
IN inventory ON inventory.inventory_id = rental.inventory_id JOIN film ON film.film_id = inv
entory.film_id WHERE DATE_PART('day', rental.return_date - rental.rental_date) > film.rental
_duration GROUP BY film.title, film.rental_duration ORDER BY title;
```

title	nbre_jour_location_autorise	nbre_de_retard
Academy Dinosaur	6	4
Ace Goldfinger	3	4
Adaptation Holes	7	1
Affair Prejudice	5	9
African Egg	6	9
Agent Truman	3	16
Airplane Sierra	6	2
Airport Pollock	6	7
Alabama Devil	3	9
Aladdin Calendar	6	6
Alamo Videotape	6	3
Alaska Phantom	6	9
Ali Forever	4	3

5. Déterminer le nombre de client par pays

-> SELECT country.country AS pays, COUNT(customer\_id) AS nbr\_client\_pays FROM customer JOIN address ON address.address\_id = customer.address\_id JOIN city ON city.city\_id = address.city\_id JOIN country ON country.country\_id = city.country\_id GROUP BY country.country ORDER BY nbr\_client\_pays DESC;

runpsql - Raccourci

```
dvdrental=# SELECT country.country AS pays, COUNT(customer_id) AS nbr_client_pays FROM custo
mer JOIN address ON address.address_id = customer.address_id JOIN city ON city.city_id = add
ress.city_id JOIN country ON country.country_id = city.country_id GROUP BY country.country O
RDER BY nbr_client_pays DESC;
```

pays	nbr_client_pays
India	60
China	53
United States	36
Japan	31
Mexico	30
Brazil	28
Russian Federation	28
Philippines	20
Turkey	15
Indonesia	14
Argentina	13
Nigeria	13
South Africa	11
Taiwan	10

6. Trouver les 5 clients qui génèrent le plus de profits pour la sociétés.

-> SELECT DISTINCT customer.first\_name, customer.last\_name, payment.customer\_id, SUM(amount) AS total\_achat FROM customer JOIN payment ON payment.customer\_id = customer.customer\_id GROUP BY customer.first\_name, customer.last\_name, payment.customer\_id ORDER BY total\_achat desc LIMIT 5;

```
runsql - Raccourci
144 | 189.60
(5 lignes)

dvdrental=# SELECT customer.first_name, customer.last_name, DISTINCT payment.customer_id, SUM(amount) AS
total_achat FROM customer JOIN payment ON payment.customer_id = customer.customer_id GROUP BY customer.
first_name, customer.last_name, payment.customer_id ORDER BY total_achat desc LIMIT 5;
ERREUR: erreur de syntaxe sur ou près de « DISTINCT »
LIGNE 1 : SELECT customer.first_name, customer.last_name, DISTINCT pay...
^
dvdrental=# SELECT DISTINCT payment.customer_id, SUM(amount) AS total_achat, customer.first_name, custom
er.last_name FROM customer JOIN payment ON payment.customer_id = customer.customer_id GROUP BY customer.
first_name, customer.last_name, payment.customer_id ORDER BY total_achat desc LIMIT 5;
 customer_id | total_achat | first_name | last_name
-----+-----+-----+-----
148 | 211.55 | Eleanor | Hunt
526 | 208.58 | Karl | Seal
178 | 194.61 | Marion | Snyder
137 | 191.62 | Rhonda | Kennedy
144 | 189.60 | Clara | Shaw
(5 lignes)

dvdrental=#
```

7. Quel est le tarif de location moyen pour chaque genre ? (du plus élevé au plus bas)

-> SELECT category.name AS genre, AVG(film.rental\_rate) AS moyenne FROM film JOIN film\_category ON film\_category.film\_id = film.film\_id JOIN category ON category.category\_id = film\_category.category\_id GROUP BY genre ORDER BY moyenne desc;

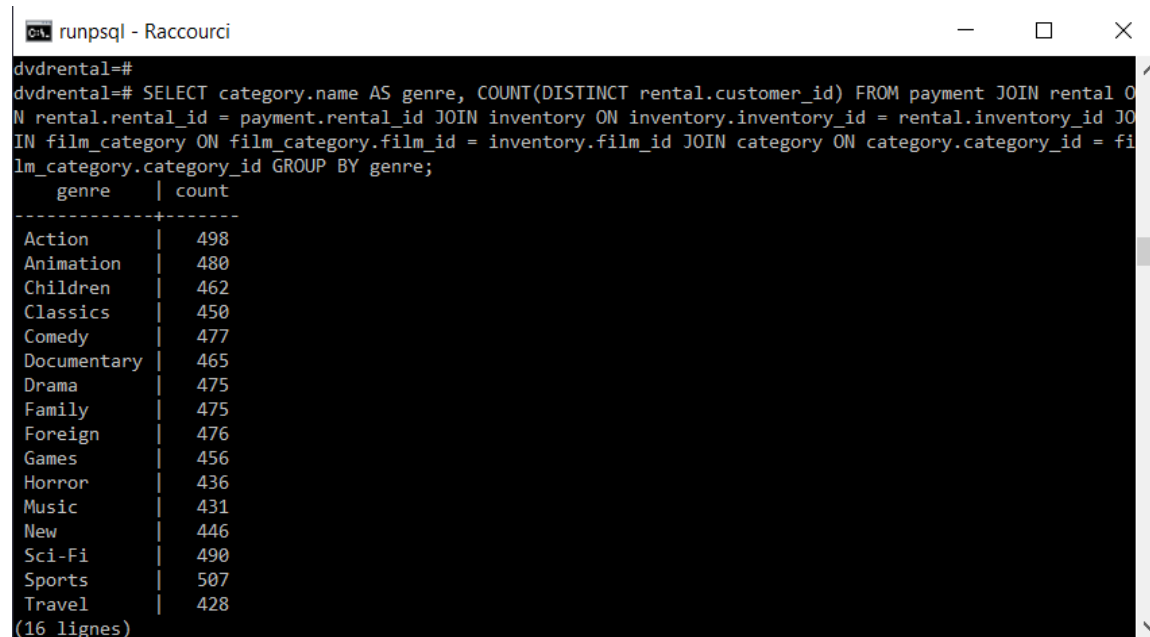
```
SQL Shell (psql)
Action | 510
(1 ligne)

dvdrental=#
dvdrental=# SELECT category.name AS genre, AVG(film.rental_rate) AS moyenne
dvdrental=# FROM film
dvdrental=# JOIN film_category
dvdrental=# ON film_category.film_id = film.film_id
dvdrental=# JOIN category
dvdrental=# ON category.category_id = film_category.category_id
dvdrental=# GROUP BY genre
dvdrental=# ORDER BY moyenne desc;
 genre | moyenne
-----+-----
Games | 3.2522950819672131
Travel | 3.2356140350877193
Sci-Fi | 3.2195081967213115
Comedy | 3.1624137931034483
Sports | 3.1251351351351351
New | 3.1169841269841270
Foreign | 3.0995890410958904
Horror | 3.0257142857142857
Drama | 3.0222580645161290
Music | 2.9507843137254902
Children | 2.8900000000000000
Animation | 2.8081818181818182
Family | 2.7581159420289855
Classics | 2.7443859649122807
Documentary | 2.6664705882352941
Action | 2.6462500000000000
(16 lignes)

dvdrental=#
```

8. Peut-on savoir combien d'utilisateurs distincts ont loué chaque genre?

-> `SELECT category.name AS genre, COUNT(DISTINCT rental.customer_id) FROM payment JOIN rental ON rental.rental_id = payment.rental_id JOIN inventory ON inventory.inventory_id = rental.inventory_id JOIN film_category ON film_category.film_id = inventory.film_id JOIN category ON category.category_id = film_category.category_id GROUP BY genre;`



```
runpsql - Raccourci
dvdrental=#
dvdrental=# SELECT category.name AS genre, COUNT(DISTINCT rental.customer_id) FROM payment JOIN rental ON
rental.rental_id = payment.rental_id JOIN inventory ON inventory.inventory_id = rental.inventory_id JO
IN film_category ON film_category.film_id = inventory.film_id JOIN category ON category.category_id = fi
lm_category.category_id GROUP BY genre;
 genre | count
-----+-----
 Action |    498
 Animation |    480
 Children |    462
 Classics |    450
 Comedy |    477
 Documentary |    465
 Drama |    475
 Family |    475
 Foreign |    476
 Games |    456
 Horror |    436
 Music |    431
 New |    446
 Sci-Fi |    490
 Sports |    507
 Travel |    428
(16 lignes)
```

9. Combien de films loués ont été retournés tard, tôt et à temps ?

-> `SELECT COUNT(CASE WHEN rental_duration > DATE_PART('day', return_date - rental_date) THEN 'Retourne en avance' END) AS en_avance, COUNT(CASE WHEN rental_duration = DATE_PART('day', return_date - rental_date) THEN 'Rendu a temps' END) AS a_temps, COUNT(CASE WHEN rental_duration < DATE_PART('day', return_date - rental_date) THEN 'Retourne en retard' END) AS en_retard FROM film JOIN inventory ON film.film_id = inventory.film_id JOIN rental ON inventory.inventory_id = rental.inventory_id;`

C:\> Sélection runpsql - Raccourci

```
dvdrental=# SELECT COUNT(CASE WHEN rental_duration > DATE_PART('day', return_date - rental_
date) THEN 'Retourne en avance' END) AS en_avance, COUNT(CASE WHEN rental_duration = DATE_PA
RT('day', return_date - rental_date) THEN 'Rendu a temps' END) AS a_temps, COUNT(CASE WHEN r
ental_duration < DATE_PART('day', return_date - rental_date) THEN 'Retourne en retard' END)
AS en_retard FROM film JOIN inventory ON film.film_id = inventory.film_id JOIN rental ON inv
entory.inventory_id = rental.inventory_id;
 en_avance | a_temps | en_retard
-----+-----+-----
       7738 |      1720 |       6403
(1 ligne)
```

dvdrental=#