

3.6: Summarizing and Cleaning Data

Answers 3.6

1. **Check for and clean dirty data:** Find out if the film table and the customer table contain any dirty data, specifically non-uniform or duplicate data, or missing values. Create a new “Answers 3.6” document and copy-paste your queries. Next to each query, write 2 to 3 sentences explaining how you would clean the data (even if the data is not dirty).

-Duplicate Data for film table:

The screenshot shows a SQL query editor with a query to check for duplicate data in the film table. The query is as follows:

```
1 SELECT film_id, title,
2        description, release_year,
3        language_id, rental_duration,
4        rental_rate, length,
5        replacement_cost, rating,
6        last_update, special_features,
7        COUNT(*)
8 FROM film
9 GROUP BY film_id, title, description,
10          release_year, language_id, rental_duration,
11          rental_rate, length,
12          replacement_cost, rating,
13          last_update, special_features
14 HAVING COUNT(*) > 1; --no result set means we have no duplicates
```

The right-hand side of the editor shows the table schema for the film table:

film_id	title	description	release_year
[PK] integer	character varying (255)	text	integer

-Duplicate Data for customer table:

The screenshot shows a SQL query editor with a query to check for duplicate data in the customer table. The query is as follows:

```
1 SELECT DISTINCT customer_id, store_id,
2                first_name, last_name,
3                email, address_id,
4                activebool, active,
5                COUNT(*)
6 FROM customer
7 GROUP BY customer_id, store_id,
8          first_name, last_name,
9          email, address_id,
10         activebool, active
11 HAVING COUNT(*) > 1 --no result set means we have no duplicates
```

The right-hand side of the editor shows the table schema for the customer table:

customer_id	store_id	first_name	last_name
[PK] integer	smallint	character varying (45)	character varying (45)

There is no returned duplicate value.

There are no returned duplicate values in both tables.

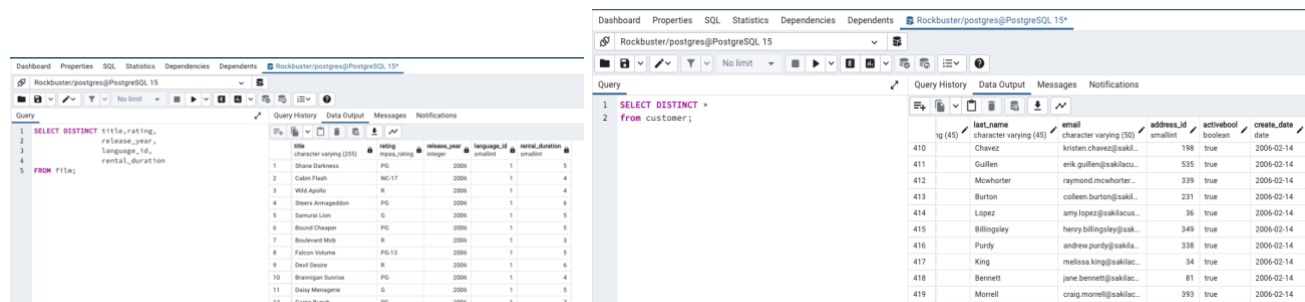
Non-uniform for customer and film tables:

I didn't see any non-uniform columns on these two tables

There are a few ways of dealing with **duplicate values**. If we have permission to alter the database, we could create a view and select only the unique records or delete the identical form from the table or view. If we cannot alter the database, you can use GROUP BY or DISTINCT to select unique records.

There were also no **non-uniform values** when I scanned through the tables with the DISTINCT statement for both tables. If we encounter one, let's say in the rating column, as we saw in the example in the exercise, we can create an UPDATE query that would make all of the results uniform.

UPDATE film SET rating = 'G' WHERE rating IN ('gen','g', 'General')

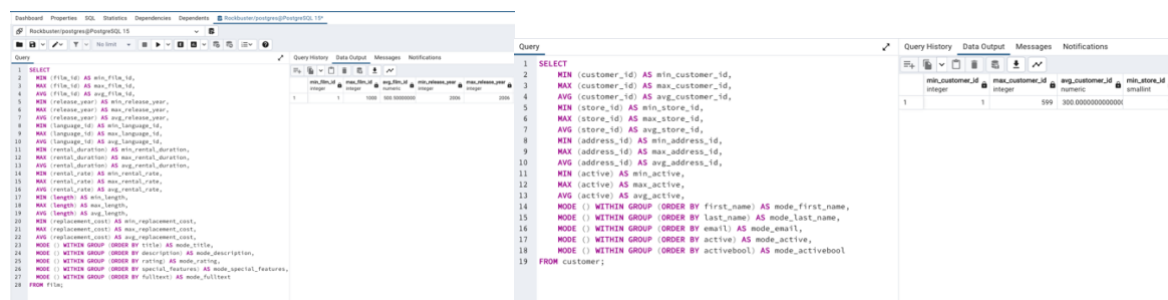


The screenshot shows a PostgreSQL dashboard with two queries. The first query is a SELECT DISTINCT statement on the film table, and the second query is a SELECT DISTINCT statement on the customer table. Both queries show their results in a table view.

id	title	rating	release_year	language_id	rental_duration
1	Shrek	PG	2001	1	5
2	Cabin in the Woods	PG	2001	1	4
3	Wild Apple	R	2001	1	4
4	Steers Unleashed	PG	2001	1	6
5	Samurai Lion	G	2001	1	5
6	Bound Cheaper	PG	2001	1	5
7	Boulevard Mob	R	2001	1	3
8	Falcon Volume	PG-13	2001	1	5
9	Devil Dearest	R	2001	1	6
10	Bringing Down the House	PG	2001	1	4
11	Daddy's Home	G	2001	1	5
12	Curse of the Hat	PG	2001	1	7

id	last_name	email	address_id	activebool	create_date
410	Chavez	christen.chavez@sakil...	198	true	2006-02-14
411	Gullien	erik.gullien@sakila...	535	true	2006-02-14
412	Mowbray	raymond.mowbray@sakila...	339	true	2006-02-14
413	Burton	colleen.burton@sakila...	231	true	2006-02-14
414	Lopez	amy.lopez@sakila...	36	true	2006-02-14
415	Billingsley	henry.billingsley@sakila...	349	true	2006-02-14
416	Purdy	andrew.purdy@sakila...	338	true	2006-02-14
417	King	melissa.king@sakila...	34	true	2006-02-14
418	Bennett	jane.bennett@sakila...	81	true	2006-02-14
419	Morell	craig.morell@sakila...	393	true	2006-02-14

2. Summarize your data: Use SQL to calculate descriptive statistics for the film and customer table. This means finding the minimum, maximum, and average values for numerical columns. For non-numerical columns, calculate the mode value. Copy-paste your SQL queries and their outputs into your answers document.



The screenshot shows a PostgreSQL dashboard with two queries. The first query is a SELECT statement with various aggregate functions (MIN, MAX, AVG, MODE) on the film table. The second query is a SELECT statement with various aggregate functions (MIN, MAX, AVG, MODE) on the customer table. Both queries show their results in a table view.

id	title	rating	release_year	language_id	rental_duration
1	Shrek	PG	2001	1	5
2	Cabin in the Woods	PG	2001	1	4
3	Wild Apple	R	2001	1	4
4	Steers Unleashed	PG	2001	1	6
5	Samurai Lion	G	2001	1	5
6	Bound Cheaper	PG	2001	1	5
7	Boulevard Mob	R	2001	1	3
8	Falcon Volume	PG-13	2001	1	5
9	Devil Dearest	R	2001	1	6
10	Bringing Down the House	PG	2001	1	4
11	Daddy's Home	G	2001	1	5
12	Curse of the Hat	PG	2001	1	7

id	last_name	email	address_id	activebool	create_date
410	Chavez	christen.chavez@sakil...	198	true	2006-02-14
411	Gullien	erik.gullien@sakila...	535	true	2006-02-14
412	Mowbray	raymond.mowbray@sakila...	339	true	2006-02-14
413	Burton	colleen.burton@sakila...	231	true	2006-02-14
414	Lopez	amy.lopez@sakila...	36	true	2006-02-14
415	Billingsley	henry.billingsley@sakila...	349	true	2006-02-14
416	Purdy	andrew.purdy@sakila...	338	true	2006-02-14
417	King	melissa.king@sakila...	34	true	2006-02-14
418	Bennett	jane.bennett@sakila...	81	true	2006-02-14
419	Morell	craig.morell@sakila...	393	true	2006-02-14

**** I will put Queries at the end of the last page****

- 2. Reflect on your work:** Back in Achievement 1, you learned about data profiling in Excel. Based on your previous experience, which tool (Excel or SQL) do you think is more effective for data profiling, and why? Consider their respective functions, ease of use, and speed. Write a short paragraph in the running document that you have started.

As I already mentioned in the previous Task. It depends on the amount that we deal with. When it comes to small amounts of data or a few columns, excel will quickly and efficiently categorize and analyze any data set. However, using SQL would be more efficient when working with large amounts of data. Once you learn how to write queries efficiently, you can quickly analyze and categorize data.

Queries

<pre> SELECT MIN (customer_id) AS min_customer_id, MAX (customer_id) AS max_customer_id, AVG (customer_id) AS avg_customer_id, MIN (store_id) AS min_store_id, MAX (store_id) AS max_store_id, AVG (store_id) AS avg_store_id, MIN (address_id) AS min_address_id, MAX (address_id) AS max_address_id, AVG (address_id) AS avg_address_id, MIN (active) AS min_active, MAX (active) AS max_active, AVG (active) AS avg_active, MODE () WITHIN GROUP (ORDER BY first_name) AS mode_first_name, MODE () WITHIN GROUP (ORDER BY last_name) AS mode_last_name, MODE () WITHIN GROUP (ORDER BY email) AS mode_email, MODE () WITHIN GROUP (ORDER BY active) AS mode_active, MODE () WITHIN GROUP (ORDER BY activebool) AS mode_activebool FROM customer; </pre>	<pre> SELECT MIN(rental_rate) AS min_renatl_rate, MAX(rental_rate) AS max_rental_rate, AVG(rental_rate) AS avg_renatal_rate, MIN(rental_duration) AS min_rental_duration, MAX(rental_duration) AS max_rental_duration, AVG(rental_duration) AS avg_rental_duration, MIN(film_id) AS min_film, MAX(film_id) AS max_film, AVG(film_id) AS avg_film, MIN(language_id) AS min_language, MAX(language_id) AS max_language, AVG(language_id) AS avg_language, MIN(length) AS min_length, MAX(length) AS max_length, AVG(length) AS avg_length, MIN(replacement_cost) AS min_replacement_cost, MAX(replacement_cost) AS max_replacement_cost, AVG(replacement_cost) AS avg_replacement_cost, MODE() WITHIN GROUP (ORDER BY rating) AS rating_value, MODE() WITHIN GROUP (ORDER BY special_features) AS feature_value, MODE() WITHIN GROUP (ORDER BY release_year) AS release_year, MODE() WITHIN GROUP (ORDER BY title) AS title_value, MODE() WITHIN GROUP (ORDER BY fulltext) AS fulltext FROM film </pre>
--	---