In this task, you'll calculate some descriptive statistics using the MIN, MAX, AVG, COUNT, SUM, and MODE() aggregates discussed in this Exercise, and you'll reflect on what you learned about data profiling back in Exercise 1.5: Data Profiling & Integrity.

Directions

Rockbuster's database engineers have loaded some new data into the database, and your manager has asked you to clean and profile it. Follow the instructions below to complete their request:

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 into it. Next to each query write 2 to 3 sentences explaining how you would clean the data (even if the data is not dirty).

FILM TABLE

I looked for duplicate data in 'film'

Used this query:

```
SELECT film_id,
title,
description,
release_year,
language_id,
rental_duration,
rental_rate,
length,
replacement_cost,
rating,
last_update,
special_features,
fulltext,
    count(*)
FROM film
GROUP BY film_id,
title,
```

```
    description,
    release_year,
    language_id,
    rental_duration,
    rental_rate,
    length,
    replacement_cost,
    rating,
    last_update,
    special_features,
    fulltext
    HAVING COUNT(*) > 1
```

And found no duplicates.

Then I looked for any unusual values – in columns that should have consistent data:

Query	**Result**
SELECT DISTINCT release_year FROM film	1 value: OK
SELECT DISTINCT language_id FROM film	1 value: OK
SELECT DISTINCT rental_duration FROM film ORDER BY rental_duration ASC	Values 3 – 7 inclusive
SELECT DISTINCT rental_rate FROM film ORDER BY rental_rate ASC	3 values
SELECT DISTINCT length FROM film ORDER BY length ASC	140 values – all same format, nothing looks out of place
SELECT DISTINCT replacement_cost FROM film ORDER BY replacement_cost ASC	21 values – all same format, nothing looks out of place
SELECT DISTINCT rating FROM film	5 values – what we expect

Initially, I looked at all these columns together, but this produces 100s of rows, because there are hundreds of unique combinations.

I looked for NULL values with this query:

```
SELECT
    COUNT(film_id) AS film_count,
    COUNT(title) AS title_count,
```

```
    COUNT(description) AS description_count,
    COUNT (release_year) AS year_count,
    COUNT (language_id) AS language_count,
    COUNT(rental_duration) AS rental_duration_count,
    COUNT(rental_rate) AS rental_rate_count,
    COUNT (length) AS length_count,
    COUNT (replacement_cost) AS replacement_cost_count,
    COUNT(rating) AS rating_count,
    COUNT (last_update) AS last_update_count,
    COUNT (special_features) AS special_features_count,
    COUNT (fulltext) AS fulltext_count
FROM film;
```

And I got the same value – 1000 – from every column, so there are no Null values.

CUSTOMER TABLE

I looked for duplicate data in 'customer'

Used this query – on only the columns that should NOT have duplicate data

```
SELECT customer_id,
email,
address_id,
    count(*)
FROM customer
GROUP BY customer_id,
email,
address_id
```

HAVING COUNT(*) > 1And found no duplicates.

Then I looked for any unusual values – in columns that should have consistent data:

Query	**Result**
SELECT DISTINCT store_id FROM customer	2 values: OK
SELECT DISTINCT activebool FROM customer	1 value: OK
SELECT DISTINCT create_date FROM customer	1 value: OK
SELECT DISTINCT last_update FROM customer	1 value: OK
SELECT DISTINCT active FROM customer	2 values: OK – but would expect to match activebool

By running this query:

SELECT active, activebool FROM customer WHERE active = 0

I was able to identify that in 15 instances active = 0 where activebool = true

I'd need to investigate more to find out which is most likely to be correct (active should be 1 or activebool should be false)

If I got a definitive answer, then I could create a VIEW and assign the correct value.

Without a definitive answer, as this is a small number of values, I might ignore these values for any analysis of active customers,

I looked for NULL values with this query:

```
SELECT
        COUNT(customer_id) AS customer_count,
        COUNT(store_id) AS store_count,
        COUNT(first_name) AS first_name_count,
        COUNT (last_name) AS last_name_count,
        COUNT (email) AS email_count,
        COUNT(address_id) AS address_count,
        COUNT(activebool) AS activebool_count,
        COUNT (create_date) AS create_date_count,
        COUNT (last_update) AS last_update_count,
        COUNT(active) AS active_count
FROM customer;
```

And I got the same value – 599– from every column, so there are no Null values.

If I had found duplicates, then I would crate a a VIEW with non-duplicate records only, and work with this view of the data.

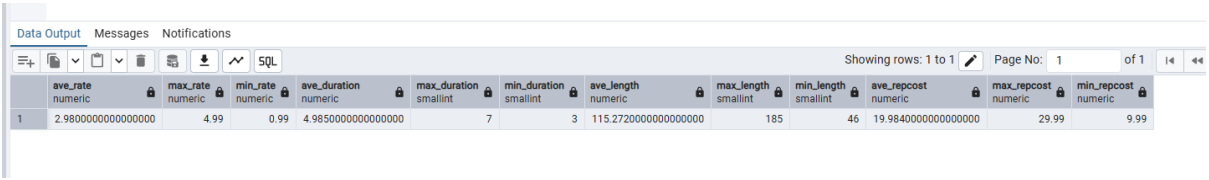
If I had found null values, then I would have looked at the whole row first.

If rows contain a lot of null values, but only small numbers of rows, I would likely continue the analysis in a view without those rows

If there were null values only in a column with numerical values, then I would see if it was possible to impute the missing values, by adding the average or median value each time a value was missing. I would do this only if a small number of values were missing.

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

```
SELECT
AVG (rental_rate) AS ave_rate,
MAX (rental_rate) AS max_rate,
MIN (rental_rate) AS min_rate,
AVG (rental_duration) AS ave_duration,
MAX (rental_duration) AS max_duration,
MIN (rental_duration) AS min_duration,
AVG (length) AS ave_length,
MAX (length) AS max_length,
MIN (length) AS min_length,
AVG (replacement_cost) AS ave_repcost,
MAX (replacement_cost) AS max_repcost,
MIN (replacement_cost) AS min_repcost
FROM film
```

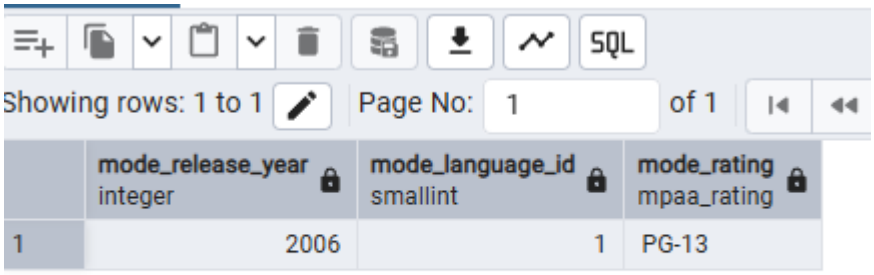


Data Output | Messages | Notifications

Showing rows: 1 to 1 | Page No: 1 of 1

	ave_rate numeric	max_rate numeric	min_rate numeric	ave_duration numeric	max_duration smallint	min_duration smallint	ave_length numeric	max_length smallint	min_length smallint	ave_repcost numeric	max_repcost numeric	min_repcost numeric
1	2.9800000000000000	4.99	0.99	4.9850000000000000	7	3	115.2720000000000000	185	46	19.9840000000000000	29.99	9.99

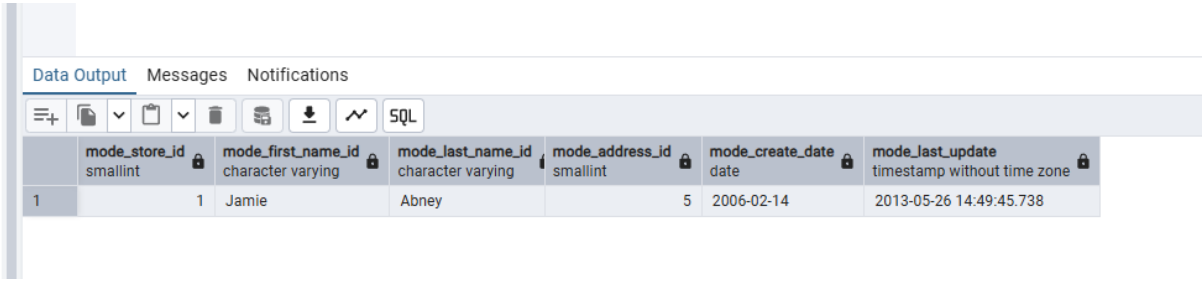
```
SELECT
  MODE() WITHIN GROUP (ORDER BY release_year) AS mode_release_year,
  MODE() WITHIN GROUP (ORDER BY language_id) AS mode_language_id,
  MODE() WITHIN GROUP (ORDER BY rating) AS mode_rating
FROM film;
```



	mode_release_year integer	mode_language_id smallint	mode_rating mpaa_rating
1	2006	1	PG-13

There aren't any numerical values in 'customer' that it makes sense to look at AVG, MAX or MIN for.

```
SELECT
    MODE() WITHIN GROUP (ORDER BY store_id) AS mode_store_id,
    MODE() WITHIN GROUP (ORDER BY first_name) AS mode_first_name_id,
    MODE() WITHIN GROUP (ORDER BY last_name) AS mode_last_name_id,
    MODE() WITHIN GROUP (ORDER BY address_id) AS mode_address_id,
    MODE() WITHIN GROUP (ORDER BY create_date) AS mode_create_date,
    MODE() WITHIN GROUP (ORDER BY last_update) AS mode_last_update
FROM customer
```



	mode_store_id smallint	mode_first_name_id character varying	mode_last_name_id character varying	mode_address_id smallint	mode_create_date date	mode_last_update timestamp without time zone
1	1	Jamie	Abney	5	2006-02-14	2013-05-26 14:49:45.738

3. **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.

At this second, I'm more confident still doing these kinds of operations in Excel. I could, for instance, create an average formula for a column and just drag it across all the columns I needed. It would be a lot quicker.
I'm surprised that using SQL it doesn't "suggest" the tables or columns you can pick from – currently I'm doing a lot of manual typing and I'm a terrible typist.

But again, as data gets bigger and more complex, I can see that using SQL will make things far easier – and therefore more reliable.