Answers 3.6

1.

```
SELECT film_id,
       title,
       description,
       release_year,
       language_id,
       rental_duration,
       length,
       replacement_cost,
       rating,
       COUNT (*)
FROM film
GROUP BY film_id,
         title,
         description,
         release_year,
         language_id,
         rental_duration,
         length,
         replacement_cost,
         rating
HAVING COUNT (*) > 1;
```

https://docs.google.com/spreadsheets/d/1gArRus4EIzsXHoyYwkjaPqiqvqnz3OvJ2uw2BOgvuMA/edit#gid=577753074

Based on the results, there is no duplicated data. If I had permission to run the commands in the database, I could use UPDATE, DELETE or CREATE VIEW commands. However, because I do not have permission, I used GROUP BY commands to select unique records.

2.

```
SELECT customer_id,
       store_id,
       first_name,
       last_name,
       email,
       address_id,
       COUNT (*)
FROM customer
GROUP BY customer_id,
         store_id,
         first_name,
         last_name,
         email,
         address_id
```

```
HAVING COUNT (*) > 1;
```

https://docs.google.com/spreadsheets/d/1kJgcZOfmB0bnUJPkd0-9mNZp4ZlxR_obC8skPtEigsk/edit#gid=1763350956

3.

```
SELECT MIN (film_id) AS min_film_id,
       MAX (film_id) AS max_film_id,
       AVG (film_id) AS avg_film_id,
       MIN (language_id) AS min_language_id,
       MAX (language_id) AS max_language_id,
       AVG (language_id) AS avg_language_id,
       MIN (rental_duration) AS min_rental_duration,
       MAX (rental_duration) AS max_rental_duration,
       AVG (rental_duration) AS avg_rental_duration,
       MIN (rental_rate) AS min_rental_rate,
       MAX (rental_rate) AS max_rental_rate,
       AVG (rental_rate) AS avg_rental_rate,
       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
FROM film;
```

https://docs.google.com/spreadsheets/d/1-xE6p8AaMxEtCbCaK622HOBWYPpcj2oysQ1_-XPRNJk/edit#gid=909839783

```
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
FROM customer;
```

```
SELECT mode() WITHIN GROUP (ORDER BY title)
       AS title,
       mode() WITHIN GROUP (ORDER BY description)
       AS description,
       mode() WITHIN GROUP (ORDER BY rating)
```

```
    AS rating,
    mode() WITHIN GROUP (ORDER BY release_year)
    AS release_year
FROM film;

SELECT mode() WITHIN GROUP (ORDER BY first_name)
    AS first_name,
    mode() WITHIN GROUP (ORDER BY last_name)
    AS last_name,
    mode() WITHIN GROUP (ORDER BY email)
    AS email
FROM customer;
```

https://docs.google.com/spreadsheets/d/1k_VV2fMsgXWaD_udFxWv9oMkIHXSObN1qsv2v7zLyGw/edit#gid=464504975

For the small data set, using Excel could be more useful. However, I do not think I will have lots of chances to profile small sizes of data sets. Because of this reason, and SQL's convenience which is once the code has been typed, it saves a huge amount of time until I get the results, I would say SQL is more effective for profiling data.