SQL exercise

Contents

Create a query for every question. Put your answers in a new document.

1. Show all events in dialog_show with regulation_id '1'.

```
SELECT *
FROM dialog_show
WHERE regulation_id = 1
```

2. Show all events in dialog_show with regulation_id '1', from March 2020.

```
SELECT *
FROM dialog_show
WHERE regulation_id = 1  AND DATE(daystamp) >= '2020-03-01'
```

3. Count the number of events from March 2020 in dialog_show per regulation.

```
SELECT regulation_id AS 'Regulation Type',
        COUNT(event) AS 'Number of Events'
FROM dialog_event
WHERE DATE(daystamp) >= '2020-03-01'
GROUP BY regulation_id
```

4. Show all unique events from dialog_event in March 2020 ordered alphabetically.

```
SELECT event
FROM dialog_event
WHERE DATE(daystamp) BETWEEN '20200301' AND '20200331'
GROUP BY event
```

5. Show top 10 locales from dialog_close with the most close counts in March 2020.

```
SELECT locale, SUM(count) AS 'Closed Counts'
FROM dialog_close
WHERE DATE(daystamp) BETWEEN '20200301' AND '20200331'
GROUP BY locale
ORDER BY SUM(count) DESC
LIMIT 10
```

6. Show bottom 10 locales from dialog_close where the summed close counts in March 2020 were at least 100 or more.

```
SELECT locale, SUM(count) AS 'Closed Counts'
FROM dialog_close
WHERE DATE(daystamp) BETWEEN '20200301' AND '20200331'
GROUP BY locale
HAVING SUM(count) >= 100
```

```
ORDER BY SUM(count)
LIMIT 10
```

7. Show all events from dialog_event, but instead of app_id, regulation_id, origin_id, flow_id and run_counter_id show the human readable names found in the corresponding tables.

```
SELECT dialog_event.id,
        dialog_event.daystamp,
        app.name AS 'app_version',
        dialog_event.app_version,
        dialog_event.os_version,
        dialog_event.locale,
        regulation.name AS 'regulation',
        origin.name AS 'origin',
        flow.name AS 'flow',
        run_counter.name AS 'run_counter',
        dialog_event.event,
        dialog_event.count
FROM dialog_event
LEFT JOIN app ON app.id = dialog_event.app_id
LEFT JOIN regulation ON regulation.id = dialog_event.regulation_id
LEFT JOIN origin ON origin.id = dialog_event.origin_id
LEFT JOIN flow ON flow.id = dialog_event.flow_id
LEFT JOIN run_counter ON run_counter.id = dialog_event.run_counter_id
```

8. (Bonus) Show the summed counts per run_counter ordered from high to low, for all run_counters but 'old'. Combine 'firstRun', 'secondRun' and 'thirdRun' under one new group called 'firstToThirdRunCount'. (Hint: Use case statement)

```
(SELECT run_counter.name AS 'run_counter_name',
        SUM(count) AS count
FROM dialog_event
LEFT JOIN run_counter ON run_counter.id = dialog_event.run_counter_id
WHERE run_counter_id != '1'
GROUP BY run_counter_id)
UNION
(SELECT "firsToThirdRunCount" AS 'run_counter_name',
        SUM(count) AS count
FROM dialog_event
LEFT JOIN run_counter ON run_counter.id = dialog_event.run_counter_id
WHERE run_counter_id = '2' OR run_counter_id = '3' OR run_counter_id = '4')
```