In the first part of Query we used this to correcting pollution data with LIKE and String operations :

```
CREATE TABLE
md_water_services.well_pollution_copy
AS (
SELECT
*
FROM
md_water_services.well_pollution
);
```

We will get a copy of well_pollution called well_pollution_copy. Now we can make the changes, and if we discover there is a

mistake in our code, we can just delete this table, and run it again.

So if we now run our query:

```
UPDATE
well_pollution_copy
SET
description = 'Bacteria:E. coli'

WHERE
description = 'Clean Bacteria:E. coli';

UPDATE
well_pollution_copy
SET
description = 'Bacteria: Giardia Lamblia'

WHERE
description = 'Clean Bacteria: Giardia Lamblia';
```

```
UPDATE
well_pollution_copy
SET
results = 'Contaminated: Biological'

WHERE
biological > 0.01 AND results = 'Clean';
```

We can then check if our errors are fixed using a SELECT query on the well_pollution_copy table:

```
SELECT
*
FROM
well_pollution_copy
WHERE
description LIKE "Clean_%"
OR (results = "Clean" AND biological > 0.01);
```

Then if we're sure it works as intended, we can change the table back to the well_pollution and delete the well_pollution_copy

table.

```
UPDATE
well_pollution_copy
SET
description = 'Bacteria: E. coli'
WHERE
description = 'Clean Bacteria: E. coli';
UPDATE
well_pollution_copy
```

```
SET
description = 'Bacteria: Giardia Lamblia'
WHERE
description = 'Clean Bacteria: Giardia Lamblia';
UPDATE
well_pollution_copy
SET
results = 'Contaminated: Biological'
WHERE
biological > 0.01 AND results = 'Clean';
DROP TABLE
md_water_services.well_pollution_copy;
```

##

IN The Second part of Query :We have to update the database again with these email addresses, so before we do, let's use a SELECT query to get the format right, then use

UPDATE and SET to make the changes.

First up, let's remove the space between the first and last names using REPLACE(). You can try this:

```
SELECT
REPLACE(employee_name, ' ','.') -- Replace the space with a full stop
FROM
employee
```

Then we can use LOWER() with the result we just got. Now the name part is correct.

```
SELECT
LOWER(REPLACE(employee_name, ' ','.')) -- Make it all lower case
FROM
employee
```

We then use CONCAT() to add the rest of the email address:

```
SELECT
CONCAT(
LOWER(REPLACE(employee_name, ' ', '.')), '@ndogowater.gov') AS new_email –– add it all together
FROM
employee
```

Quick win! Since you have done this before, you can go ahead and UPDATE the email column this time with the email addresses. Just make sure to

check if it worked!

```
UPDATE employee
SET email = CONCAT(LOWER(REPLACE(employee_name, ' ', '.')),

'@ndogowater.gov')
```

To filter a row we use WHERE, but using CASE() in SELECT can filter columns. We can use a CASE() function for each day to separate the queue

time column into a column for each day. Let's begin by only focusing on Sunday. So, when a row's DAYNAME(time_of_record) is Sunday, we

make that value equal to time_in_queue, and NULL for any days.

If you run this query you will see what I mean.

```
SELECT
TIME_FORMAT(TIME(time_of_record), '%H:00') AS hour_of_day,
DAYNAME(time_of_record),
CASE
WHEN DAYNAME(time_of_record) = 'Sunday' THEN time_in_queue
ELSE NULL
END AS Sunday
```

```
FROM
visits
WHERE
time_in_queue != 0; -- this exludes other sources with 0 queue times.
```

By adding AVG() around the CASE() function, we calculate the average, but since all of the other days' values are 0, we get an average for Sunday

only, rounded to 0 decimals. To aggregate by the hour, we can group the data by hour_of_day, and to make the table chronological, we also order

by hour_of_day.

This is the form of the query we will use:

```
SELECT
TIME_FORMAT(TIME(time_of_record), '%H:00') AS hour_of_day,
-- Sunday
ROUND(AVG(
CASE
WHEN DAYNAME(time_of_record) = 'Sunday' THEN time_in_queue
ELSE NULL
END
),0) AS Sunday,
-- Monday
ROUND(AVG(
CASE
WHEN DAYNAME(time_of_record) = 'Monday' THEN time_in_queue
ELSE NULL
END
),0) AS Monday
-- Tuesday
```

```
-- Wednesday
FROM
visits
WHERE
time_in_queue != 0 -- this excludes other sources with 0 queue times
GROUP BY
hour_of_day
ORDER BY
hour_of_day;
```