SQLite(DBeaver)

The first step I did is to load the db. file into the DBeaver and start to take a look at the data. From the raw data, it has 6965 rows.
After having a general understanding of the data, I start to check for duplicate data.

```
SELECT * FROM exchange_transactions et
```

```
SELECT COUNT (DISTINCT TRANSACTION_id) from exchange_transactions et
SELECT COUNT(TRANSACTION_id) from exchange_transactions et
```

There are a few duplicate primary keys (transcation_id) in the data sets with duplicate rows. Since transcation_id is a uniquely generated serial number, and the row associated with the duplicate primary key is also duplicate, it is safe to drop out those rows.
Now we have 6915 rows

```
DELETE FROM exchange_transactions
WHERE TRANSACTION_id in
(SELECT max(TRANSACTION_id) FROM exchange_transactions group by TRANSACTION_id
HAVING COUNT(TRANSACTION_id) > 1)
```

My next step is to check Null & NA data, I found out there are null data in parent_transcation_id and room_name. Parent_transcation by description tells the null is normal when a response is not approved, deny and mark updated.
I found the 2559 Room_name outside of 6915 is empty when the action is deny_request, mark_updated, and request. My understanding is when room_name is empty, the transaction is either in processing (if approved it will give room_name on another transaction) or request denied.

```
SELECT "action",count("action") FROM exchange_transactions et
WHERE room_name  is NULL
GROUP BY "action"
```

After that, the data was cleaned. The next step I did is find the number of approved requests, releases, and transfers.

```
SELECT "action", COUNT("action") FROM exchange_transactions et
GROUP BY "action"
ORDER BY COUNT("action") DESC
```

I am also interested in how many possible surgery times we created since we start to use the iQueue by calculating the end_time, and start_time difference when the action is approve_transfer and released. Under those two actions, the surgeons give the OR time slot to other surgeons. Therefore, the hospital can make more surgeries by using Google Big Query

Google Big Query

```
SELECT COUNT(action) as total_transfer,sum(TIME_DIFF(end_time, start_time,MINUTE)) as
Total_time_tran
FROM `LeanTaaS.iQueue`
WHERE action = "APPROVE_TRANSFER"
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	total_transfer	Total_time
1	255	124435

```
SELECT COUNT(action) as total_release,sum(TIME_DIFF(end_time, start_time,MINUTE)) as
Total_time_rele
FROM `LeanTaaS.iQueue`
WHERE action = "RELEASE"
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	total_release	Total_time_r...
1	1199	574125

I added up the total time saved 124435+574125 = 698520 minutes and divide by 60 to 11642 hours. By assuming the average surgery time is 8 hours, the hospital approximately can create 1455 more surgeries.

After the initial analysis, I start to think about how to increase adoption. I start using **Tableau** to create some visualizations of the surgeon's usage of iQueue patterns. Asking myself which surgeon/room has less transfer, and why? Maybe the system is difficult to use or the current scheduling is perfect and doesn't need further adjustments? Why is the denial rate for specific rooms and locations higher than others?