25.JAN.2025

DATA RELATED TO REGIMENS CLEANING AND VALIDATION

1. OVERIVEW

- OBJECTIVE
  Ensure data quality for Sact_Regimen and related tables to establish a foundation for successful analysis

- ANALYSIS SCOPE
  Includes the Sact_Regimen, AV_Patient, AV_Tumour, AV_Gene, Sact_Cycle, Sact_Outcome, and Sact_Drug_Detail tables.

2. KEY TABLES AND FIELDS WITH RATIONALE

- **SACT_REGIMEN(TREATMENT REGIMEN DATA)**

  - MERGED_REGIMEN_ID
  : Unique ID for regimens, essential for data tracking and differentiation.

  - ENCORE_PATIENT_ID
  : Foregin key linking regimens to patients

  - START_DATE_OF_REGIMEN, DATE_DECISION_TO_TREAT:
  : Crucial for analyzing reatment duration and modification patterns.

  - MAPPED_REGIMEN
  : Regimen name, essential for comparison and success rate analysis.

- **AV_PATIENT(PATIENT INFORMATION)**

  - PATIENTID
  : Unique ID for patients, foundational for data integration and analysis.

  - GENDER
  : Important for analyzing gender-specific treatment responses

  - VITALSTATUS, VITALSTATUSDATE
  : Critical for survival rate analysis and treatment outcome evaluation

- **AV_TUMOUR(TUMOR INFORMATION)**

  - TUMOURID
  : Unique ID for tumors, necessary for tumor-specific analysis.

  - DIAGNOSISDATEBEST
  : Diagnosis date, comparable to treatment start date.

  - STAGE_BEST
  : Tumor stage, correlates with treatment success.

- **AV_GENE(GENE MUTATION DATA)**

  - GENEID
  : Unique ID for genes.

  - ABNORMAL_GAT
  : Abnormal gene data influencing treatment efficacy.

- **SACT_CYCLE, SACT_OUTCOME, SACT_DRUG_DETAIL(SUPPORTING DATA)**

  - provide details on treatment cycles, outcomes, and drug administration, which are critical for detailed analysis and understanding treatment success.

3. EXCLUDED ELEMENTS

- **HEIGHT AND WIEIGHT**
  - Height_at_start_of_regimen and Weight_at_start_of_regimen fields have a high volume of missing values and are less likely to directly impact the analysis.

- **RADIOTHERAPY TABLES**
  - Radiotherapy-related tables(e.g, Rtds_combined, Rtds_Prescription) are excluded as the analysis focuses on chemotherapy regimens.

4. DATA VALIDATION STRATEGY

- **NULL VALUE DETECTION AND HANDLING:**

  - Identify and replace missing values in critical fields.

- **DUPLICATE DATA DETECTION AND REMOVAL:**

  - Eliminate duplicates in key ID fields.

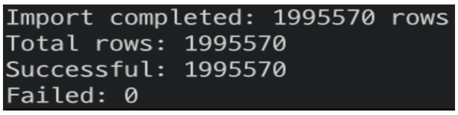
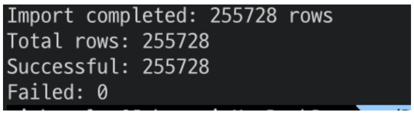
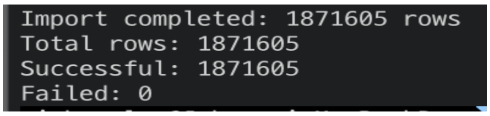
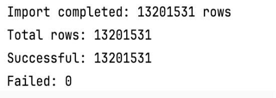
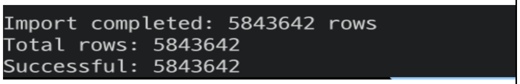
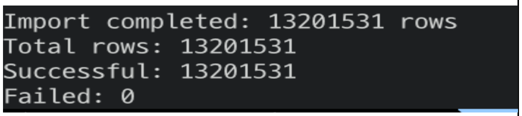
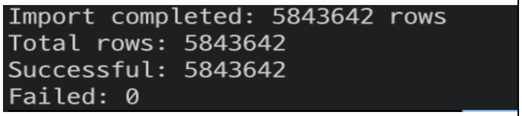
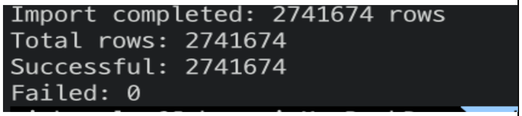
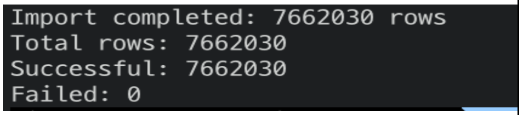
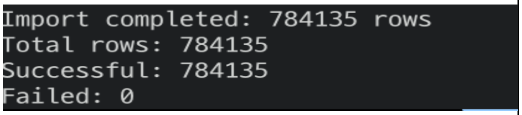
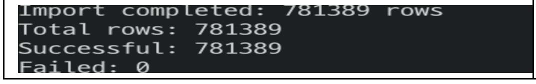
- **FOREIGN KEY INTERGIRTY CHECK:**

  - Verify table relationships to ensure consistency and acracy.

- **DATE FILED VALIDATION:**

  - Validate treatment durations and diagnosis dates.

5. WHETHER THE GIVEN CSV HAS BEEN FULLY IMPORTED INTO THE DATABASE

ii. sim_av_tumour	*iii. sim_av_gene*
 Import completed: 1995570 rows Total rows: 1995570 Successful: 1995570 Failed: 0	 Import completed: 255728 rows Total rows: 255728 Successful: 255728 Failed: 0
iv. sim_av_patient	*v. sim_rtds_combined*
 Import completed: 1871605 rows Total rows: 1871605 Successful: 1871605 Failed: 0	 Import completed: 13201531 rows Total rows: 13201531 Successful: 13201531 Failed: 0
vi. sim_rtds_episode	*vii. sim_rtds_exposure*
 Import completed: 5843642 rows Total rows: 5843642 Successful: 5843642	 Import completed: 13201531 rows Total rows: 13201531 Successful: 13201531 Failed: 0
viii. sim_rtds_prescription	*ix. sim_sact_cycle*
 Import completed: 5843642 rows Total rows: 5843642 Successful: 5843642 Failed: 0	 Import completed: 2741674 rows Total rows: 2741674 Successful: 2741674 Failed: 0
x. sim_sact_drug_detail	*xi. sim_sact_outcome*
 Import completed: 7662030 rows Total rows: 7662030 Successful: 7662030 Failed: 0	 Import completed: 784135 rows Total rows: 784135 Successful: 784135 Failed: 0
xii. sim_sact_regimen	
 Import completed: 781389 rows Total rows: 781389 Successful: 781389 Failed: 0	

6. TO CHECK FOR MISSING VVALUES IN CRTICAL FIELDS AND VALIDATE DATE INTERGITY FOCUSING ON FOREGIN KEY RELATIONSHIPS.

A. SACT_REGIMEN

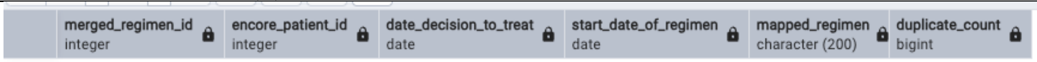
Check for missing values in critical fields of SACT_REGIMEN table:

```
SELECT COUNT(*)
FROM SACT_REGIMEN
WHERE MERGED_REGIMEN_ID IS NULL
OR ENCORE_PATIENT_ID IS NULL
  OR DATE_DECISION_TO_TREAT IS NULL
  OR START_DATE_OF_REGIMEN IS NULL
  OR MAPPED_REGIMEN IS NULL;
```

Result: 76199

ii. Check for duplicate records in SACT_REGIMEN:

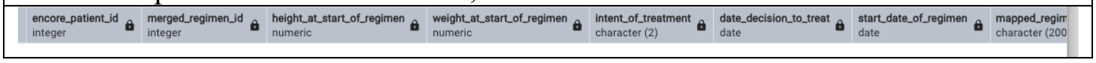
```
SELECT
   MERGED_REGIMEN_ID,
   ENCORE_PATIENT_ID,
   DATE_DECISION_TO_TREAT,
   START_DATE_OF_REGIMEN,
   MAPPED_REGIMEN,
   COUNT(*) AS duplicate_count
FROM SACT_REGIMEN
GROUP BY
   MERGED_REGIMEN_ID,
   ENCORE_PATIENT_ID,
   DATE_DECISION_TO_TREAT,
   START_DATE_OF_REGIMEN,
   MAPPED_REGIMEN
HAVING COUNT(*) > 1;
```



merged_regimen_id	encore_patient_id	date_decision_to_treat	start_date_of_regimen	mapped_regimen	duplicate_count
integer	integer	date	date	character (200)	bigint

iii. Relationship validation between SACT_REGIMEN and AV_PATIENT:

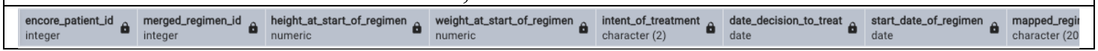
```
SELECT sr.*
FROM SACT_REGIMEN sr
LEFT JOIN AV_Patient ap ON sr.ENCORE_PATIENT_ID = ap.PATIENTID
WHERE ap.PATIENTID IS NULL;
```



encore_patient_id	merged_regimen_id	height_at_start_of_regimen	weight_at_start_of_regimen	intent_of_treatment	date_decision_to_treat	start_date_of_regimen	mapped_regim
integer	integer	numeric	numeric	character (2)	date	date	character (200

iv. Relationship validation between SACT_REGIMEN and SACT_TUMOUR:

```
SELECT sr.*
FROM SACT_REGIMEN sr
LEFT JOIN AV_Tumour at ON sr.ENCORE_PATIENT_ID = at.PATIENTID
WHERE at.PATIENTID IS NULL;
```



encore_patient_id	merged_regimen_id	height_at_start_of_regimen	weight_at_start_of_regimen	intent_of_treatment	date_decision_to_treat	start_date_of_regimen	mapped_regi
integer	integer	numeric	numeric	character (2)	date	date	character (20

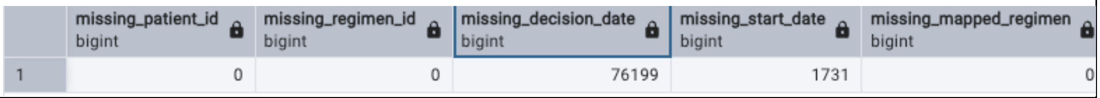
Follow-up Action

Address missing values in SACT_REGIMEN (76,199 records):

ii. *Identify frequently missing fields:*

```
SELECT
   SUM(CASE WHEN ENCORE_PATIENT_ID IS NULL THEN 1 ELSE 0 END) AS
missing_patient_id,
   SUM(CASE WHEN MERGED_REGIMEN_ID IS NULL THEN 1 ELSE 0 END) AS
missing_regimen_id,
   SUM(CASE WHEN DATE_DECISION_TO_TREAT IS NULL THEN 1 ELSE 0 END) AS
missing_decision_date,
   SUM(CASE WHEN START_DATE_OF_REGIMEN IS NULL THEN 1 ELSE 0 END) AS
missing_start_date,
   SUM(CASE WHEN MAPPED_REGIMEN IS NULL THEN 1 ELSE 0 END) AS
missing_mapped_regimen
FROM SACT_REGIMEN;
```

Result:



	missing_patient_id bigint	missing_regimen_id bigint	missing_decision_date bigint	missing_start_date bigint	missing_mapped_regimen bigint
1	0	0	76199	1731	0

2. For Date of decision to treat and Date of starting Regimen

- Time difference-based imputation

- **CASE 1 – Only DATE_DECISION_TO_TREAT exists**

Condition : DATE_DECISION_TO_TREAT exists,

START_DATE_OF_REGIMEN is missing

Solution :

```
# 1. Calculate median time difference
query_median = """
SELECT EXTRACT(DAY FROM
(start_date_of_regimen::timestamp -
date_decision_to_treat::timestamp)) as diff_days
FROM sact_regimen
WHERE date_decision_to_treat IS NOT NULL
AND start_date_of_regimen IS NOT NULL;
"""
median_df = pd.read_sql(query_median, connection)
median_diff = median_df['diff_days'].median()
```

The code targets records in the database where both date fields (DATE_DECISION_TO_TREAT and START_DATE_OF_REGIMEN) are not NULL and calculates the difference between the two dates. It then computes the **median** of these differences and uses it as the basis for imputing missing values in Case 1 and Case 2.

```
cursor.execute("""
    UPDATE sact_regimen
    SET start_date_of_regimen =
date_decision_to_treat + INTERVAL '%s days'
    WHERE date_decision_to_treat IS NOT NULL
    AND start_date_of_regimen IS NULL;
""" % int(median_diff))
```

The code updates the START_DATE_OF_REGIMEN by adding the median difference in days to the DATE_DECISION_TO_TREAT value.

- ### CASE 2 – Only START_DATE_OF_REGIMEN exists

Condition : DATE_DECISION_TO_TREAT is missing,

START_DATE_OF_REGIMEN is exists

Solution :

```
# 1. Calculate median time difference
query_median = """
SELECT EXTRACT(DAY FROM
(start_date_of_regimen::timestamp -
date_decision_to_treat::timestamp)) as diff_days
FROM sact_regimen
WHERE date_decision_to_treat IS NOT NULL
AND start_date_of_regimen IS NOT NULL;
"""
median_df = pd.read_sql(query_median, connection)
median_diff = median_df['diff_days'].median()
```

The code targets records in the database where both date fields (DATE_DECISION_TO_TREAT and START_DATE_OF_REGIMEN) are not NULL and calculates the difference between the two dates. It then computes the **median** of these differences and uses it as the basis for imputing missing values in Case 1 and Case 2.

```
# Case 2: Only start_date_of_regimen exists
cursor.execute("""
    UPDATE sact_regimen
    SET date_decision_to_treat =
start_date_of_regimen - INTERVAL '%s days'
    WHERE date_decision_to_treat IS NULL
    AND start_date_of_regimen IS NOT NULL;
""" % int(median_diff))
```

The code updates the DATE_DECISION_TO_TREAT by subtracting the median difference in days from the START_DATE_OF_REGIMEN value.

- ### CASE 3 – Both dates are missing

Condition : Both DATE_DECISION_TO_TREAT and TART_DATE_OF_REGIMEN are missing

Solution :

```
# Case 3: Both dates are missing
cursor.execute("""
    UPDATE sact_regimen
    SET
        date_decision_to_treat = %s::date,
        start_date_of_regimen = %s::date +
INTERVAL '7 days'
    WHERE date_decision_to_treat IS NULL
    AND start_date_of_regimen IS NULL;
""", (mid_date, mid_date))
```

To process the data, the code retrieves the minimum value (earliest_date) and maximum value (latest_date) of the START_DATE_OF_REGIMEN field from the database and calculates the median date (mid_date).

```
query_case3 = f"""
UPDATE SACT_REGIMEN
SET
    DATE_DECISION_TO_TREAT = '{mid_date}'::date,
    START_DATE_OF_REGIMEN = '{mid_date}'::date +
INTERVAL '7 days'
WHERE DATE_DECISION_TO_TREAT IS NULL
AND START_DATE_OF_REGIMEN IS NULL;"""
```

The code sets both DATE_DECISION_TO_TREAT and START_DATE_OF_REGIMEN to the median date (mid_date) and calculates the START_DATE_OF_REGIMEN as 7 days after the mid_date.

Positive impacts:

- Increased Analytical Capability:
  Imputing missing values preserves dataset size and increases the number of analyzable data points.

- Restored Temporal Relationships:
  Enables analysis of treatment delays, regimen modification patterns, and comparisons of treatment initiation items.

- Improved Model Performance:
  A dataset without missing values provides more stable and reliable training data for machine learning models.

Potential Negative impacts:

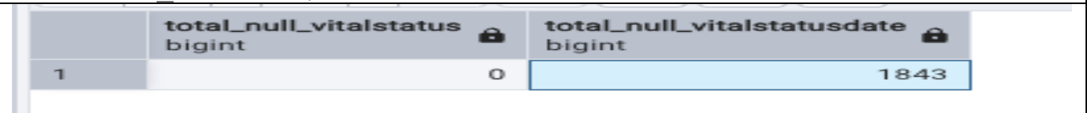
- Risk of Data Distortion in CASE 3:
  Applying a simple mid_date and +7 days rule to all CASE 3 records may not accurately reflect the real data distribution.

- Limitations of Median-Based Imputation:
  Using the same median value for all records may fail to capture unique characteristics in specific groups.

B. AV_PATIENT

i. Check for missing values in critical fields of AV_PATIENT table:

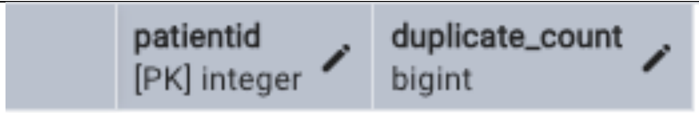
```
SELECT
   SUM(CASE WHEN VITALSTATUS IS NULL THEN 1 ELSE 0 END) AS
total_null_vitalstatus,
   SUM(CASE WHEN VITALSTATUSDATE IS NULL THEN 1 ELSE 0 END)
AS total_null_vitalstatusdate
FROM AV_PATIENT;
```



	total_null_vitalstatus bigint	total_null_vitalstatusdate bigint
1	0	1843

ii. Check for duplicate records in AV_PATIENT:

```
SELECT
   PATIENTID,
   COUNT(*) AS duplicate_count
FROM AV_PATIENT
GROUP BY PATIENTID
HAVING COUNT(*) > 1;
```



	patientid [PK] integer	duplicate_count bigint

Follow-up Action

```
query = """
SELECT
    ap.patientid,
    PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY EXTRACT(EPOCH
FROM sr.start_date_of_regimen)) AS median_start_date,
    PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY EXTRACT(EPOCH
FROM sr.date_decision_to_treat)) AS median_decision_date
FROM av_patient ap
LEFT JOIN sact_regimen sr ON ap.patientid = sr.encore_patient_id
GROUP BY ap.patientid;
"""
```

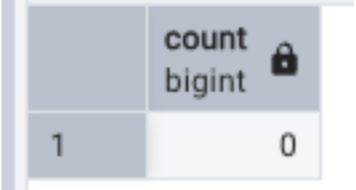
Calculate Medians:
PERCENTILE_CONT(0.5) computes the median (50th percentile) for the given column. It returns the median of start_date_of_regimen and date_decision_to_treat for each patientid.
Group by Patient:
The query groups data by patientid using GROUP BY so that each patient has one row in the output containing their median values.

After the imputation process, there are still 710 missing values. These records likely do not have a foreign key relationship with sact_regimen. Considering the small number of these values and their negligible impact on the analysis, it is reasonable to remove them from the dataset.

C. AV_TUMOUR

i. Check for missing values in critical fields of AV_TUMOUR table:

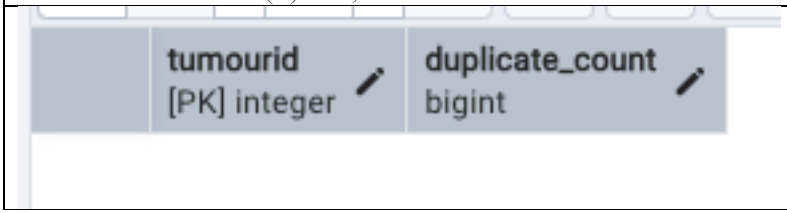
```
SELECT COUNT(*)
FROM AV_TUMOUR
WHERE TUMOURID IS NULL
  OR DIAGNOSISDATEBEST IS NULL
  OR DIAGNOSISDATEBEST IS NULL ;
```



	count bigint
1	0

ii. Check for duplicate records in AV_TUMOUR:

```
SELECT
   TUMOURID,
   COUNT(*) AS duplicate_count
FROM AV_TUMOUR
GROUP BY TUMOURID
HAVING COUNT(*) > 1;
```

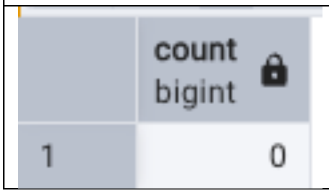


	tumourid [PK] integer	duplicate_count bigint

D. AV_GENE

Check for missing values in critical fields of AV_GENE table:

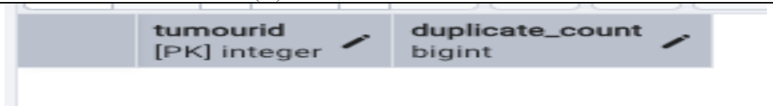
```
SELECT COUNT(*)
FROM AV_TUMOUR
WHERE TUMOURID IS NULL
  OR DIAGNOSISDATEBEST IS NULL
  OR DIAGNOSISDATEBEST IS NULL ;
```



	count bigint
1	0

ii. Check for duplicate records in AV_GENE:

```
SELECT
   GENEID,
   COUNT(*) AS duplicate_count
FROM AV_GENE
GROUP BY GENEID
HAVING COUNT(*) > 1;
```



	tumourid [PK] integer	duplicate_count bigint

E. Support data

```
SACT_CYCLE Table Checks:
Nulls in MERGED_REGIMEN_ID: 0
Nulls in CYCLE_NUMBER: 0
Nulls in START_DATE_OF_CYCLE: 6842

SACT_OUTCOME Table Checks:
Nulls in MERGED_REGIMEN_ID: 0
Nulls in REGIMEN_OUTCOME_SUMMARY: 0

SACT_DRUG_DETAIL Table Checks:
Nulls in MERGED_DRUG_DETAIL_ID: 0
Nulls in ACTUAL_DOSE_PER_ADMINISTRATION: 79543
Nulls in ADMINISTRATION_DATE: 18116
Database connection closed.
```

Nulls in START_DATE_OF_CYCLE: 6842

Based on the `START_DATE_OF_REGIMEN`, set the date of the missing data to the same value.

```
UPDATE sact_cycle sc
SET start_date_of_cycle = sr.start_date_of_regimen
FROM sact_regimen sr
WHERE sc.merged_regimen_id = sr.merged_regimen_id
  AND sc.start_date_of_cycle IS NULL;
```

Nulls in ACTUAL_DOSE_PER_ADMINISTRATION: 79543

Since the total amount of data is sufficiently large, removing these entries would not have a significant impact on the overall dataset size

```
DELETE FROM sact_drug_detail
WHERE actual_dose_per_adminstration IS NULL;
```

Nulls in ADMINISTRATION_DATE: 18116

Since the total amount of data is sufficiently large, removing these entries would not have a significant impact on the overall dataset size

```
DELETE FROM sact_drug_detail
WHERE actual_dose_per_adminstration IS NULL;
```