About the Author

This report was authored by Ayoub Majjid, a fifth-year computer engineering student at EMSI with a background in Experimental Sciences. His academic journey has provided a strong foundation in mathematics, physics, and chemistry, and has shaped a growing expertise in technology, system design, and data engineering.

Ayoub currently serves as Tech Lead and Entrepreneur at Intellcap, where he leads three innovation projects focused on building impactful and scalable startup solutions. His work emphasizes transforming ideas into robust technical systems, with a particular interest in data platforms, system architecture, and end-to-end engineering workflows.

Email: ayoub@majjid.com 🌐 Website: https://majjid.com

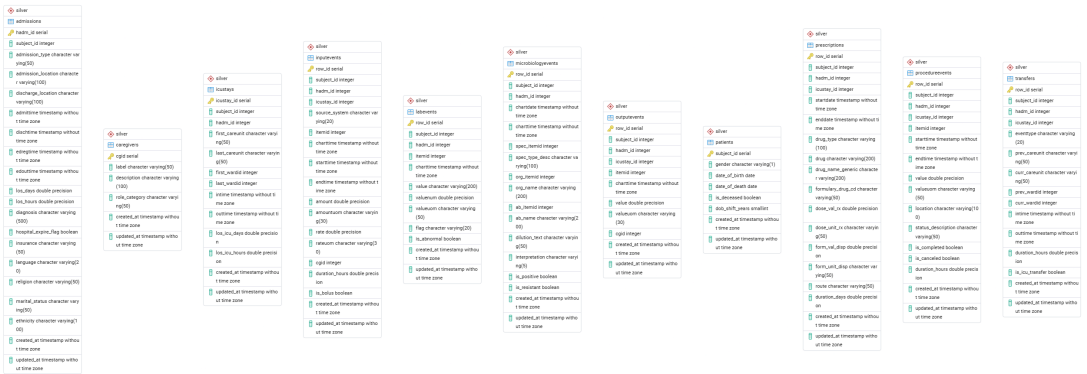
Copyright & Disclaimer

This report is provided for **educational and professional demonstration purposes**. The MIMIC-III dataset remains the property of its respective owners and is used in accordance with applicable data usage agreements.

No patient-identifiable information is disclosed or altered in this document.

🥈 Silver Layer Analysis

This notebook analyzes the **Silver Layer** of the MIMIC-III Data Warehouse.



What is the Silver Layer?

The Silver layer contains **cleaned and standardized data** transformed from Bronze:

- Data types are consistent and validated

- Derived fields are calculated (LOS, age, flags)
- Null handling and default values applied
- Ready for dimensional modeling

In [14]:
```
# Setup
import sys
sys.path.insert(0, '../../..')

import pandas as pd
from sqlalchemy import text
from app.shared import get_db

pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
```

In [15]:
```
# Helper functions
def query_df(sql, limit=10):
    with get_db() as session:
        result = session.execute(text(sql))
        df = pd.DataFrame(result.fetchall(), columns=result.keys())
    return df.head(limit) if limit else df

def table_count(schema, table):
    with get_db() as session:
        return session.execute(text(f"SELECT COUNT(*) FROM {schema}.{table}")).
```

---

1. PATIENTS Table (Silver)

Transformations from Bronze:

- Age at first admission calculated
- Mortality flag derived from death dates
- Age group categorization

Column	Type	Description	Transformation
subject_id	INTEGER	**PK** - Patient ID	Direct copy
gender	VARCHAR	Gender	Direct copy
dob	TIMESTAMP	Date of birth	Direct copy
dod	TIMESTAMP	Date of death	Direct copy
expire_flag	BOOLEAN	Is deceased	**Derived**: dod IS NOT NULL
age_at_first_admit	FLOAT	Age at first admission	**Calculated**

In [16]:
```
print(f"Total Patients (Silver): {table_count('silver', 'patients'):,}")
query_df("SELECT * FROM silver.patients ORDER BY subject_id LIMIT 5")
```

```
2026-01-01 17:43:36,840 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:36,841 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.patients
2026-01-01 17:43:36,842 INFO sqlalchemy.engine.Engine [cached since 152.2s ago]
{}
2026-01-01 17:43:36,844 INFO sqlalchemy.engine.Engine COMMIT
Total Patients (Silver): 100
2026-01-01 17:43:36,848 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:36,849 INFO sqlalchemy.engine.Engine SELECT * FROM silver.pati
ents ORDER BY subject_id LIMIT 5
2026-01-01 17:43:36,850 INFO sqlalchemy.engine.Engine [cached since 152.2s ago]
{}
2026-01-01 17:43:36,854 INFO sqlalchemy.engine.Engine COMMIT
```

Out[16]:

	subject_id	gender	date_of_birth	date_of_death	is_deceased	dob_shift_years	created_at	updated_at
0	10006	F	2094-03-05	2165-08-12	True	None	2026-01-01 12:11:42.339197	2026-01-01 12:11:42.339197
1	10011	F	2090-06-05	2126-08-28	True	None	2026-01-01 12:11:42.339197	2026-01-01 12:11:42.339197
2	10013	F	2038-09-03	2125-10-07	True	None	2026-01-01 12:11:42.339197	2026-01-01 12:11:42.339197
3	10017	F	2075-09-21	2152-09-12	True	None	2026-01-01 12:11:42.339197	2026-01-01 12:11:42.339197
4	10019	M	2114-06-20	2163-05-15	True	None	2026-01-01 12:11:42.339197	2026-01-01 12:11:42.339197

In [17]:

```
# Mortality analysis
query_df("""
    SELECT
        gender,
        COUNT(*) as total,
        SUM(CASE WHEN is_deceased THEN 1 ELSE 0 END) as deceased,
        ROUND(
    (100.0 * SUM(CASE WHEN is_deceased THEN 1 ELSE 0 END) / COUNT(*))::numeric,
    1
) AS mortality_rate
    FROM silver.patients
    GROUP BY gender
""", limit=None)
```

```
2026-01-01 17:43:36,876 INFO sqlalchemy.engine.Engine BEGIN
(implicit) 2026-01-01 17:43:36,878 INFO sqlalchemy.engine.Engine SELECT
gender, COUNT() as total, SUM(CASE WHEN is_deceased THEN 1 ELSE 0
END) as deceased, ROUND( (100.0  SUM(CASE WHEN is_deceased THEN 1
ELSE 0 END) / COUNT(*))::numeric, 1 ) AS mortality_rate FROM
silver.patients GROUP BY gender
```

2026-01-01 17:43:36,879 INFO sqlalchemy.engine.Engine [cached since 146.6s ago] {} 2026-01-01 17:43:36,883 INFO sqlalchemy.engine.Engine COMMIT

Out[17]:

	gender	total	deceased	mortality_rate
0	M	45	45	100.0
1	F	55	55	100.0

---

2. ADMISSIONS Table (Silver)

Transformations from Bronze:

- Length of stay calculated in days and hours
- Hospital expire flag standardized as boolean
- Date fields normalized

Column	Type	Description	Transformation
hadm_id	INTEGER	**PK** - Admission ID	Direct copy
subject_id	INTEGER	Patient ID	Direct copy
admittime	TIMESTAMP	Admission time	Direct copy
dischtime	TIMESTAMP	Discharge time	Direct copy
los_days	FLOAT	Length of stay (days)	**Calculated**: dischtime - admittime
los_hours	FLOAT	Length of stay (hours)	**Calculated**: los_days * 24
hospital_expire_flag	BOOLEAN	Died in hospital	**Standardized** to boolean

In [18]:
```
print(f"Total Admissions (Silver): {table_count('silver', 'admissions'):,}")
query_df("""
    SELECT hadm_id, subject_id, admission_type, admittime, dischtime,
           los_days, los_hours, hospital_expire_flag
    FROM silver.admissions
    ORDER BY hadm_id LIMIT 5
""")
```

```
2026-01-01 17:43:36,901 INFO sqlalchemy.engine.Engine BEGIN
(implicit) 2026-01-01 17:43:36,902 INFO sqlalchemy.engine.Engine SELECT
COUNT(*) FROM silver.admissions 2026-01-01 17:43:36,903 INFO
sqlalchemy.engine.Engine [cached since 146.5s ago] {} 2026-01-01
17:43:36,908 INFO sqlalchemy.engine.Engine COMMIT Total Admissions
(Silver): 129 2026-01-01 17:43:36,911 INFO sqlalchemy.engine.Engine
BEGIN (implicit) 2026-01-01 17:43:36,912 INFO sqlalchemy.engine.Engine
SELECT hadm_id, subject_id, admission_type, admittime, dischtime,
los_days, los_hours, hospital_expire_flag FROM silver.admissions ORDER
BY hadm_id LIMIT 5
```

2026-01-01 17:43:36,913 INFO sqlalchemy.engine.Engine [cached since 146.5s ago] {} 2026-01-01 17:43:36,918 INFO sqlalchemy.engine.Engine COMMIT

Out[18]:

	hadm_id	subject_id	admission_type	admittime	dischtime	los_days	los_hours	hospital_expire_flag
0	100375	10056	EMERGENCY	2129-05-02 00:12:00	2129-05-06 13:40:00	4.56	109.47	False
1	100969	42430	EMERGENCY	2142-11-26 21:20:00	2142-11-30 10:55:00	3.57	85.58	True
2	101361	41914	EMERGENCY	2145-12-01 18:13:00	2145-12-18 17:45:00	16.98	407.53	False
3	102203	42135	EMERGENCY	2127-07-23 15:21:00	2127-08-04 16:00:00	12.03	288.65	False
4	103379	44228	EMERGENCY	2170-12-15 03:14:00	2170-12-24 18:00:00	9.62	230.77	False

In [19]:

```
# LOS statistics by admission type
query_df("""
    SELECT
        admission_type,
        COUNT(*) as admissions,
        ROUND(AVG(los_days::numeric), 2) as avg_los_days,
        ROUND(MIN(los_days::numeric), 2) as min_los_days,
        ROUND(MAX(los_days::numeric), 2) as max_los_days
    FROM silver.admissions
    GROUP BY admission_type
    ORDER BY avg_los_days DESC
""", limit=None)
```

```
2026-01-01 17:43:36,963 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:36,965 INFO sqlalchemy.engine.Engine
    SELECT
        admission_type,
        COUNT(*) as admissions,
        ROUND(AVG(los_days::numeric), 2) as avg_los_days,
        ROUND(MIN(los_days::numeric), 2) as min_los_days,
        ROUND(MAX(los_days::numeric), 2) as max_los_days
    FROM silver.admissions
    GROUP BY admission_type
    ORDER BY avg_los_days DESC

2026-01-01 17:43:36,966 INFO sqlalchemy.engine.Engine [cached since 93.35s ago]
{}
2026-01-01 17:43:36,971 INFO sqlalchemy.engine.Engine COMMIT
```

Out[19]:

	admission_type	admissions	avg_los_days	min_los_days	max_los_days
0	ELECTIVE	8	11.67	4.09	39.70
1	EMERGENCY	119	9.23	0.04	123.98
2	URGENT	2	6.26	5.69	6.83

---

3. ICUSTAYS Table (Silver)

Transformations from Bronze:

- ICU length of stay calculated
- Care unit standardization

Column	Type	Description	Transformation
icustay_id	INTEGER	**PK** - ICU stay ID	Direct copy
subject_id	INTEGER	Patient ID	Direct copy
hadm_id	INTEGER	Admission ID	Direct copy
first_careunit	VARCHAR	First ICU unit	Direct copy
last_careunit	VARCHAR	Last ICU unit	Direct copy
intime	TIMESTAMP	ICU entry time	Direct copy
outtime	TIMESTAMP	ICU exit time	Direct copy
los_icu_days	FLOAT	ICU LOS (days)	**Calculated**
los_icu_hours	FLOAT	ICU LOS (hours)	**Calculated**

In [20]:

```
print(f"Total ICU Stays (Silver): {table_count('silver', 'icustays'):,}")
query_df("""
    SELECT icustay_id, subject_id, hadm_id, first_careunit,
           intime, outtime, los_icu_days, los_icu_hours
```

```
    FROM silver.icustays LIMIT 5
""")
```

```
2026-01-01 17:43:37,002 INFO sqlalchemy.engine.Engine BEGIN
(implicit) 2026-01-01 17:43:37,003 INFO sqlalchemy.engine.Engine SELECT
COUNT(*) FROM silver.icustays 2026-01-01 17:43:37,004 INFO
sqlalchemy.engine.Engine [generated in 0.00127s] {} 2026-01-01
17:43:37,012 INFO sqlalchemy.engine.Engine COMMIT Total ICU Stays
(Silver): 136 2026-01-01 17:43:37,018 INFO sqlalchemy.engine.Engine
BEGIN (implicit) 2026-01-01 17:43:37,019 INFO sqlalchemy.engine.Engine
SELECT icustay_id, subject_id, hadm_id, first_careunit, intime, outtime,
los_icu_days, los_icu_hours FROM silver.icustays LIMIT 5
```

2026-01-01 17:43:37,020 INFO sqlalchemy.engine.Engine [generated in 0.00101s] {} 2026-01-01 17:43:37,025 INFO sqlalchemy.engine.Engine COMMIT

Out[20]:

	icustay_id	subject_id	hadm_id	first_careunit	intime	outtime	los_icu_days	los_icu_hours
0	258147	10042	148562	CSRU	2147-02-06 12:40:59	2147-02-14 16:04:43	8.14	195.40
1	266122	10043	168674	MICU	2185-04-14 00:23:56	2185-04-16 00:44:18	2.01	48.34
2	270154	10044	124073	MICU	2152-10-03 02:02:49	2152-10-07 16:52:38	4.62	110.83
3	203766	10045	126949	MICU	2129-11-24 22:46:57	2129-12-01 06:03:55	6.30	151.28
4	213289	10046	133110	CCU	2194-07-26 23:49:00	2194-07-29 21:01:00	2.88	69.20

In [21]:

```
# ICU performance metrics
query_df("""
    SELECT
        first_careunit,
        COUNT(*) as total_stays,
        COUNT(DISTINCT subject_id) as unique_patients,
        ROUND(AVG(los_icu_days)::numeric, 2) as avg_icu_los,
        ROUND(MAX(los_icu_days)::numeric, 2) as max_icu_los
    FROM silver.icustays
    GROUP BY first_careunit
    ORDER BY total_stays DESC
""", limit=None)
```

```
2026-01-01 17:43:37,051 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,052 INFO sqlalchemy.engine.Engine
    SELECT
        first_careunit,
        COUNT(*) as total_stays,
        COUNT(DISTINCT subject_id) as unique_patients,
        ROUND(AVG(los_icu_days)::numeric, 2) as avg_icu_los,
        ROUND(MAX(los_icu_days)::numeric, 2) as max_icu_los
    FROM silver.icustays
    GROUP BY first_careunit
    ORDER BY total_stays DESC

2026-01-01 17:43:37,054 INFO sqlalchemy.engine.Engine [generated in 0.00184s]
{}
2026-01-01 17:43:37,060 INFO sqlalchemy.engine.Engine COMMIT
```

Out[21]:

	first_careunit	total_stays	unique_patients	avg_icu_los	max_icu_los
0	MICU	77	56	3.96	31.12
1	SICU	23	20	5.67	35.41
2	CCU	19	18	5.75	25.00
3	TSICU	11	11	3.59	22.39
4	CSRU	6	6	3.63	8.14

---

4. LABEVENTS Table (Silver)

Transformations from Bronze:

- Abnormal flag derived from flag column
- Numeric value validation

Column	Type	Description	Transformation
row_id	INTEGER	**PK** - Event ID	Direct copy
subject_id	INTEGER	Patient ID	Direct copy
hadm_id	INTEGER	Admission ID	Direct copy
itemid	INTEGER	Lab test ID	Direct copy
charttime	TIMESTAMP	Time of test	Direct copy
valuenum	FLOAT	Numeric value	Validated
valueuom	VARCHAR	Unit of measure	Direct copy
is_abnormal	BOOLEAN	Abnormal result	**Derived**: flag = 'abnormal'

In [22]:

```
print(f"Total Lab Events (Silver): {table_count('silver', 'labevents'):,}")
query_df("""
```

```
    SELECT row_id, subject_id, itemid, charttime, value, valuenum, valueuom, is
    FROM silver.labevents LIMIT 5
""")
```

```
2026-01-01 17:43:37,081 INFO sqlalchemy.engine.Engine BEGIN
(implicit) 2026-01-01 17:43:37,083 INFO sqlalchemy.engine.Engine SELECT
COUNT(*) FROM silver.labevents 2026-01-01 17:43:37,084 INFO
sqlalchemy.engine.Engine [generated in 0.00097s] {} 2026-01-01
17:43:37,119 INFO sqlalchemy.engine.Engine COMMIT Total Lab Events
(Silver): 76,074 2026-01-01 17:43:37,124 INFO sqlalchemy.engine.Engine
BEGIN (implicit) 2026-01-01 17:43:37,124 INFO sqlalchemy.engine.Engine
SELECT row_id, subject_id, itemid, charttime, value, valuenum, valueuom,
is_abnormal FROM silver.labevents LIMIT 5
```

2026-01-01 17:43:37,125 INFO sqlalchemy.engine.Engine [generated in 0.00073s] {} 2026-01-01 17:43:37,132 INFO sqlalchemy.engine.Engine COMMIT

Out[22]:

	row_id	subject_id	itemid	charttime	value	valuenum	valueuom	is_abnormal
0	6244575	10006	51137	2164-09-24 20:21:00	1+	NaN	None	False
1	6244593	10006	51267	2164-09-24 20:21:00	2+	NaN	None	False
2	6244594	10006	51268	2164-09-24 20:21:00	OCCASIONAL	NaN	None	False
3	6244595	10006	51274	2164-09-24 20:21:00	15.3	15.3	sec	True
4	6244596	10006	51275	2164-09-24 20:21:00	28.5	28.5	sec	False

In [23]:

```
# Abnormal rate analysis
query_df("""
    SELECT
        CASE WHEN is_abnormal THEN 'Abnormal' ELSE 'Normal' END as status,
        COUNT(*) as count,
        ROUND(100.0 * COUNT(*) / SUM(COUNT(*)) OVER(), 1) as percentage
    FROM silver.labevents
    WHERE is_abnormal IS NOT NULL
    GROUP BY is_abnormal
""", limit=None)
```

```
2026-01-01 17:43:37,155 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,156 INFO sqlalchemy.engine.Engine
    SELECT
        CASE WHEN is_abnormal THEN 'Abnormal' ELSE 'Normal' END as status,
        COUNT(*) as count,
        ROUND(100.0 * COUNT(*) / SUM(COUNT(*)) OVER(), 1) as percentage
    FROM silver.labevents
    WHERE is_abnormal IS NOT NULL
    GROUP BY is_abnormal

2026-01-01 17:43:37,157 INFO sqlalchemy.engine.Engine [generated in 0.00119s]
{}
2026-01-01 17:43:37,269 INFO sqlalchemy.engine.Engine COMMIT
```

Out[23]:

	status	count	percentage
0	Normal	46337	60.9
1	Abnormal	29737	39.1

---

5. PRESCRIPTIONS Table (Silver)

Transformations from Bronze:

- Duration calculation
- Dose normalization

Column	Type	Description	Transformation
row_id	INTEGER	**PK** - Prescription ID	Direct copy
subject_id	INTEGER	Patient ID	Direct copy
hadm_id	INTEGER	Admission ID	Direct copy
drug	VARCHAR	Drug name	Direct copy
drug_name_generic	VARCHAR	Generic name	Direct copy
startdate	TIMESTAMP	Start time	Direct copy
enddate	TIMESTAMP	End time	Direct copy
dose_unit_rx	VARCHAR	Dose unit	Direct copy
route	VARCHAR	Admin route	Direct copy

In [24]:

```
print(f"Total Prescriptions (Silver): {table_count('silver', 'prescriptions'):,
query_df("""
    SELECT row_id, subject_id, drug, drug_name_generic, startdate, enddate, rou
    FROM silver.prescriptions LIMIT 5
""")
```

```
2026-01-01 17:43:37,283 INFO sqlalchemy.engine.Engine BEGIN
(implicit) 2026-01-01 17:43:37,285 INFO sqlalchemy.engine.Engine SELECT
COUNT(*) FROM silver.prescriptions 2026-01-01 17:43:37,286 INFO
sqlalchemy.engine.Engine [generated in 0.00092s] {} 2026-01-01
17:43:37,297 INFO sqlalchemy.engine.Engine COMMIT Total Prescriptions
(Silver): 10,398 2026-01-01 17:43:37,301 INFO sqlalchemy.engine.Engine
BEGIN (implicit) 2026-01-01 17:43:37,302 INFO sqlalchemy.engine.Engine
SELECT row_id, subject_id, drug, drug_name_generic, startdate, enddate,
route FROM silver.prescriptions LIMIT 5
```

2026-01-01 17:43:37,302 INFO sqlalchemy.engine.Engine [generated in 0.00056s] {} 2026-01-01 17:43:37,307 INFO sqlalchemy.engine.Engine COMMIT

Out[24]:

	row_id	subject_id	drug	drug_name_generic	startdate	enddate	route
0	86578	43735	Carvedilol	Carvedilol	2128-11-05	2128-11-05	PO
1	455857	10104	Bisacodyl	Bisacodyl	2120-08-25	2120-08-31	PO
2	456597	10104	NS	None	2120-08-25	2120-08-27	IV
3	653017	10027	Docusate Sodium (Liquid)	Docusate Sodium (Liquid)	2190-07-13	2190-07-16	NG
4	925240	41795	Magnesium Sulfate	None	2145-09-06	2145-09-06	IV

In [25]:

```
# Most common administration routes
query_df("""
    SELECT route, COUNT(*) as count
    FROM silver.prescriptions
    WHERE route IS NOT NULL
    GROUP BY route
    ORDER BY count DESC
    LIMIT 10
""")
```

```
2026-01-01 17:43:37,323 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,324 INFO sqlalchemy.engine.Engine
    SELECT route, COUNT(*) as count
    FROM silver.prescriptions
    WHERE route IS NOT NULL
    GROUP BY route
    ORDER BY count DESC
    LIMIT 10

2026-01-01 17:43:37,325 INFO sqlalchemy.engine.Engine [generated in 0.00083s]
{}
2026-01-01 17:43:37,339 INFO sqlalchemy.engine.Engine COMMIT
```

Out[25]:

	route	count
0	IV	5312
1	PO	2545
2	IV DRIP	828
3	SC	470
4	PO/NG	262
5	IH	238
6	PR	114
7	TP	97
8	NG	94
9	IM	91

Silver Layer Summary

Key improvements from Bronze to Silver:

1. **Derived metrics**: LOS, age, abnormal flags
2. **Data quality**: Validated types, handled nulls
3. **Standardization**: Consistent boolean flags
4. **Ready for Gold**: Clean data for dimensional modeling

In [26]:

```
# Silver layer summary
silver_tables = ['patients', 'admissions', 'icustays', 'labevents', 'prescripti
                 'transfers', 'inputevents', 'outputevents', 'procedureevents',

summary = []
for table in silver_tables:
    try:
        count = table_count('silver', table)
        summary.append({'table': table, 'count': count})
    except:
        summary.append({'table': table, 'count': 'N/A'})

pd.DataFrame(summary)
```

```
2026-01-01 17:43:37,356 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,357 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.patients
2026-01-01 17:43:37,358 INFO sqlalchemy.engine.Engine [cached since 152.7s ago]
{}
2026-01-01 17:43:37,360 INFO sqlalchemy.engine.Engine COMMIT
2026-01-01 17:43:37,363 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,364 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.admissions
2026-01-01 17:43:37,366 INFO sqlalchemy.engine.Engine [cached since 147s ago]
{}
2026-01-01 17:43:37,369 INFO sqlalchemy.engine.Engine COMMIT
2026-01-01 17:43:37,372 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,373 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.icustays
2026-01-01 17:43:37,374 INFO sqlalchemy.engine.Engine [cached since 0.3703s ag
o] {}
2026-01-01 17:43:37,376 INFO sqlalchemy.engine.Engine COMMIT
2026-01-01 17:43:37,378 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,379 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.labevents
2026-01-01 17:43:37,379 INFO sqlalchemy.engine.Engine [cached since 0.2964s ag
o] {}
2026-01-01 17:43:37,387 INFO sqlalchemy.engine.Engine COMMIT
2026-01-01 17:43:37,389 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,389 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.prescriptions
2026-01-01 17:43:37,390 INFO sqlalchemy.engine.Engine [cached since 0.1053s ag
o] {}
2026-01-01 17:43:37,393 INFO sqlalchemy.engine.Engine COMMIT
2026-01-01 17:43:37,395 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,396 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.transfers
2026-01-01 17:43:37,396 INFO sqlalchemy.engine.Engine [generated in 0.00043s]
{}
2026-01-01 17:43:37,403 INFO sqlalchemy.engine.Engine COMMIT
2026-01-01 17:43:37,405 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,406 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.inputevents
2026-01-01 17:43:37,407 INFO sqlalchemy.engine.Engine [generated in 0.00046s]
{}
2026-01-01 17:43:37,423 INFO sqlalchemy.engine.Engine COMMIT
2026-01-01 17:43:37,425 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,426 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.outputevents
2026-01-01 17:43:37,426 INFO sqlalchemy.engine.Engine [generated in 0.00046s]
{}
2026-01-01 17:43:37,433 INFO sqlalchemy.engine.Engine COMMIT
2026-01-01 17:43:37,435 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,436 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.procedureevents
2026-01-01 17:43:37,437 INFO sqlalchemy.engine.Engine [generated in 0.00061s]
{}
2026-01-01 17:43:37,443 INFO sqlalchemy.engine.Engine COMMIT
2026-01-01 17:43:37,445 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,446 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.microbiologyevents
2026-01-01 17:43:37,446 INFO sqlalchemy.engine.Engine [generated in 0.00064s]
{}
2026-01-01 17:43:37,453 INFO sqlalchemy.engine.Engine COMMIT
```

```
2026-01-01 17:43:37,455 INFO sqlalchemy.engine.Engine BEGIN (implicit)
2026-01-01 17:43:37,456 INFO sqlalchemy.engine.Engine SELECT COUNT(*) FROM silv
er.caregivers
2026-01-01 17:43:37,456 INFO sqlalchemy.engine.Engine [generated in 0.00053s]
{}
2026-01-01 17:43:37,462 INFO sqlalchemy.engine.Engine COMMIT
```

Out[26]:

	table	count
0	patients	100
1	admissions	129
2	icustays	136
3	labevents	76074
4	prescriptions	10398
5	transfers	524
6	inputevents	48023
7	outputevents	11320
8	procedureevents	753
9	microbiologyevents	2003
10	caregivers	7567