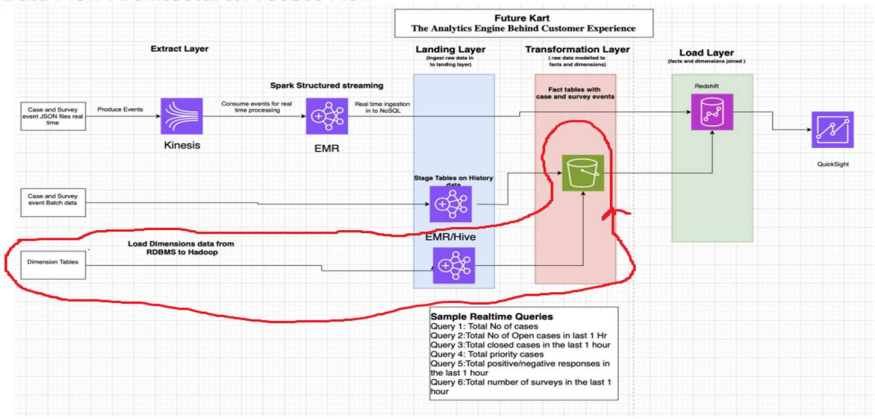
PROJECT-1 FutureCart Documentation

Name: Khushi Baurasi
Data Engineer Bootcamp 2025
Date: 31 July 2025

Historical data for Dimension Tables Step-by-Step -

3. Data Flow Architecture/Process Flow



1. Launch EC2 Instance

- Launched an **Amazon EC2 instance** (Amazon Linux 2) for initial setup.
- Configured **security groups** to allow inbound SSH (port 22) and MySQL (port 3306).
- Connected to the instance using:

ssh -i “khushi-seoul”.pem ec2-user@ ec2-3-35-79-218.ap-northeast-2.compute.amazonaws.com

---

2. Install MySQL Server

- Installed MySQL on the EC2 instance:

sudo yum update -y

sudo yum install mysql-server -y

sudo systemctl start mysqld

sudo mysql -u root -p

---

3. Create MySQL Database and Tables

CREATE DATABASE futurecart_crm;

USE futurecart_crm;

- Created 8 dimension tables:

1. futurecart_calendar_details

```
CREATE TABLE futurecart_calendar_details (
 calendar_date DATE,
 date_desc VARCHAR(50),
 week_day_nbr SMALLINT,
 week_number SMALLINT,
 week_name VARCHAR(50),
 year_week_number INT,
 month_number SMALLINT,
 month_name VARCHAR(50),
 quarter_number SMALLINT,
 quarter_name VARCHAR(50),
 half_year_number SMALLINT,
 half_year_name VARCHAR(50),
 geo_region_cd CHAR(2)
);
```

2. futurecart_call_center_details

```
CREATE TABLE futurecart_call_center_details (
 call_center_id INT PRIMARY KEY,
 call_center_name VARCHAR(100),
 location VARCHAR(100),
```

```
 manager_name VARCHAR(100),
 contact_number VARCHAR(20),
 opened_date DATE,
 is_active BOOLEAN
);
```

---

3. futurecart_case_category_details

sql

CopyEdit

```
CREATE TABLE futurecart_case_category_details (
 category_id INT PRIMARY KEY,
 category_name VARCHAR(100),
 description TEXT,
 parent_category_id INT
);
```

---

4. futurecart_case_country_details

sql

CopyEdit

```
CREATE TABLE futurecart_case_country_details (
 country_code CHAR(2) PRIMARY KEY,
 country_name VARCHAR(100),
 region VARCHAR(100),
 language VARCHAR(50)
);
```

---

5. futurecart_case_priority_details

```
CREATE TABLE futurecart_case_priority_details (
```

```
 priority_id INT PRIMARY KEY,
 priority_level VARCHAR(50),
 description TEXT,
 escalation_time_hours INT
);
```

---

6. futurecart_employee_details

```
CREATE TABLE futurecart_employee_details (
 employee_id INT PRIMARY KEY,
 first_name VARCHAR(50),
 last_name VARCHAR(50),
 department VARCHAR(100),
 designation VARCHAR(100),
 email VARCHAR(100),
 contact_number VARCHAR(20),
 hire_date DATE,
 is_active BOOLEAN
);
```

---

7. futurecart_product_details

```
CREATE TABLE futurecart_product_details (
 product_id INT PRIMARY KEY,
 product_name VARCHAR(100),
 category VARCHAR(100),
 brand VARCHAR(100),
 price DECIMAL(10,2),
 launch_date DATE,
 is_active BOOLEAN
```

);

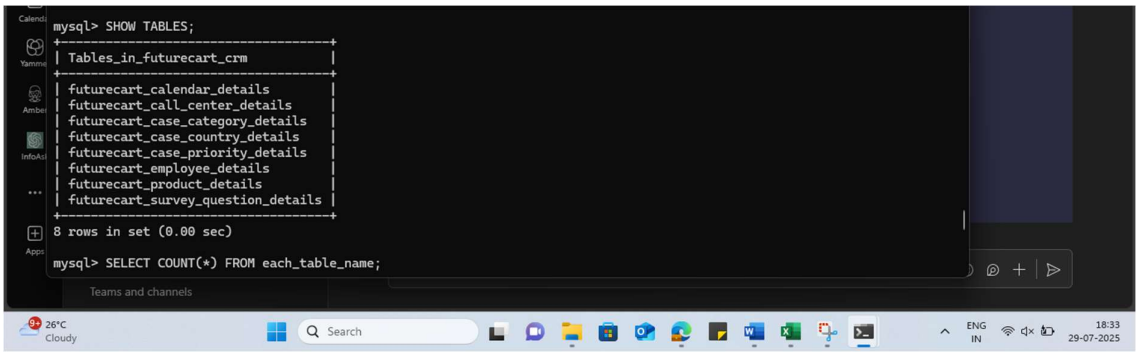
---

8. futurecart_survey_question_details

sql

CopyEdit

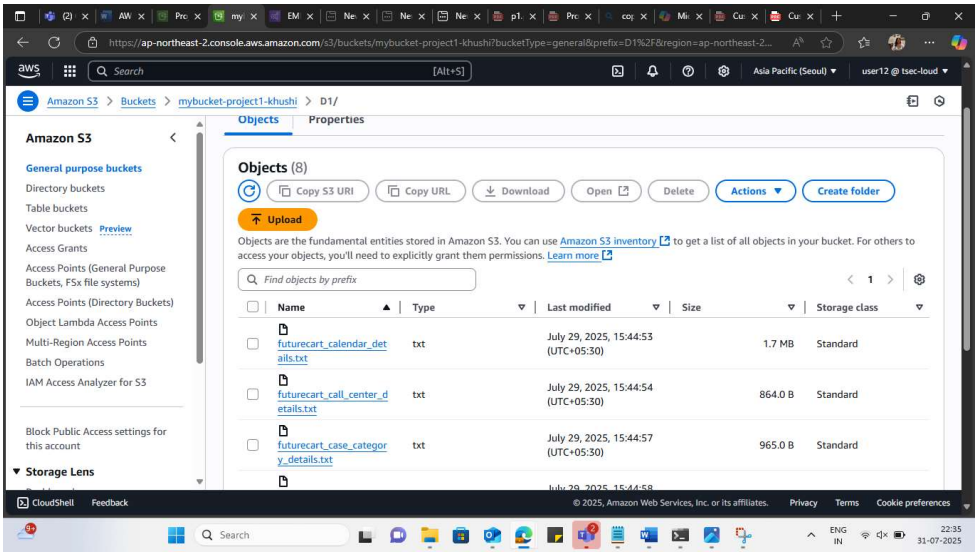
```
CREATE TABLE futurecart_survey_question_details (
 question_id INT PRIMARY KEY,
 question_text TEXT,
 question_type VARCHAR(50),  -- e.g., multiple-choice, rating, open-ended
 is_mandatory BOOLEAN,
 category VARCHAR(50)
);
```



---

4. Upload Input Files to S3

- Uploaded .txt files provided by instructor to my S3 Bucket



---

5. Move Files from S3 to EC2

- Connected to EC2 via CLI.
- Downloaded .txt files from S3 to EC2 instance:

  scp -i khushi-seoul.pem futurecart_calendar_details.txt ec2-user@ec2-3-36-70-214.ap-northeast-2.compute.amazonaws.com:~

  scp -i khushi-seoul.pem futurecart_call_center_details.txt ec2-user@ec2-3-36-70-214.ap-northeast-2.compute.amazonaws.com:~

  scp -i khushi-seoul.pem futurecart_case_category_details.txt ec2-user@ec2-3-36-70-214.ap-northeast-2.compute.amazonaws.com:~

  scp -i khushi-seoul.pem futurecart_case_country_details.txt ec2-user@ec2-3-36-70-214.ap-northeast-2.compute.amazonaws.com:~

  scp -i khushi-seoul.pem futurecart_case_priority_details.txt ec2-user@ec2-3-36-70-214.ap-northeast-2.compute.amazonaws.com:~

  scp -i khushi-seoul.pem futurecart_employee_details.txt ec2-user@ec2-3-36-70-214.ap-northeast-2.compute.amazonaws.com:~

scp -i khushi-seoul.pem futurecart_product_details.txt ec2-user@ec2-3-36-70-214.ap-northeast-2.compute.amazonaws.com:~

scp -i khushi-seoul.pem futurecart_survey_question_details.txt ec2-user@ec2-3-36-70-214.ap-northeast-2.compute.amazonaws.com:~

---

6. Load Data into MySQL

- Loaded each .txt file into the corresponding MySQL table using:

```
USE futurecart_crm;

LOAD DATA LOCAL INFILE '/home/ec2-user/futurecart_calendar_details.txt'
INTO TABLE futurecart_calendar_details
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n';
IGNORE 1 LINES;

LOAD DATA LOCAL INFILE '/home/ec2-user/futurecart_calendar_details.txt'
INTO TABLE futurecart_calendar_details
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n';
IGNORE 1 LINES;

LOAD DATA LOCAL INFILE '/home/ec2-user/futurecart_case_category_details.txt'
INTO TABLE futurecart_case_category_details
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n'
```

```
IGNORE 1 LINES;



TRUNCATE TABLE futurecart_case_country_details;
LOAD DATA LOCAL INFILE '/home/ec2-user/futurecart_case_country_details.txt'
INTO TABLE futurecart_case_country_details
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;

LOAD DATA LOCAL INFILE '/home/ec2-user/futurecart_case_priority_details.txt'
INTO TABLE futurecart_case_priority_details
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 LINES;

LOAD DATA LOCAL INFILE '/home/ec2-user/futurecart_employee_details.txt'
INTO TABLE futurecart_employee_details
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 LINES;

LOAD DATA LOCAL INFILE '/home/ec2-user/futurecart_product_details.txt'
INTO TABLE futurecart_product_details
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 LINES;
```

LOAD DATA LOCAL INFILE '/home/ec2-user/futurecart_survey_question_details.txt'

INTO TABLE futurecart_survey_question_details

FIELDS TERMINATED BY '\t'

LINES TERMINATED BY '\n'

IGNORE 1 LINES;

---

7. Launch Amazon EMR Cluster

- Launched a new **EMR Cluster** with the following:
    - Applications: Hadoop, Hive, Spark
    - Cluster name: futurecart-emr-cluster-khushi
- Created new EC2 instance for CLI access.

---

8. Connect to EMR Master Node

- Used EMR master node's SSH endpoint to connect via CLI:

ssh -i "khushi-seoul.pem" ec2-user@ec2-3-35-79-218.ap-northeast-2.compute.amazonaws.com

---

9. Install JDBC Driver and Set Up Hive

- Downloaded **MySQL JDBC driver**:

Sudo wget https://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-8.0.33.tar.gz

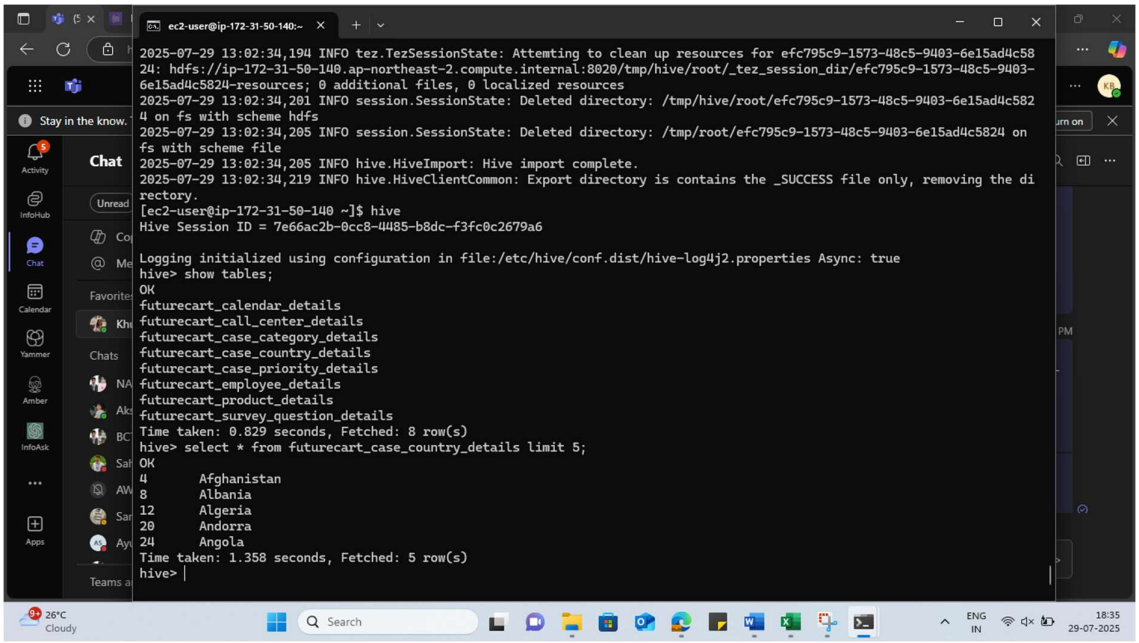
Sudo tar -xvzf mysql-connector-java-8.0.33.tar.gz

cp mysql-connector-java-8.0.33/mysql-connector-java-8.0.33.jar /usr/lib/hive/lib/

---

10. Import Data from MySQL to Hive

- Created Hive tables matching MySQL schema.
- Imported MySQL data into Hive using:

- 

---

11. Export Hive Data to S3 Using Spark

- Wrote and ran a **PySpark job** to export Hive tables to CSV format in S3:

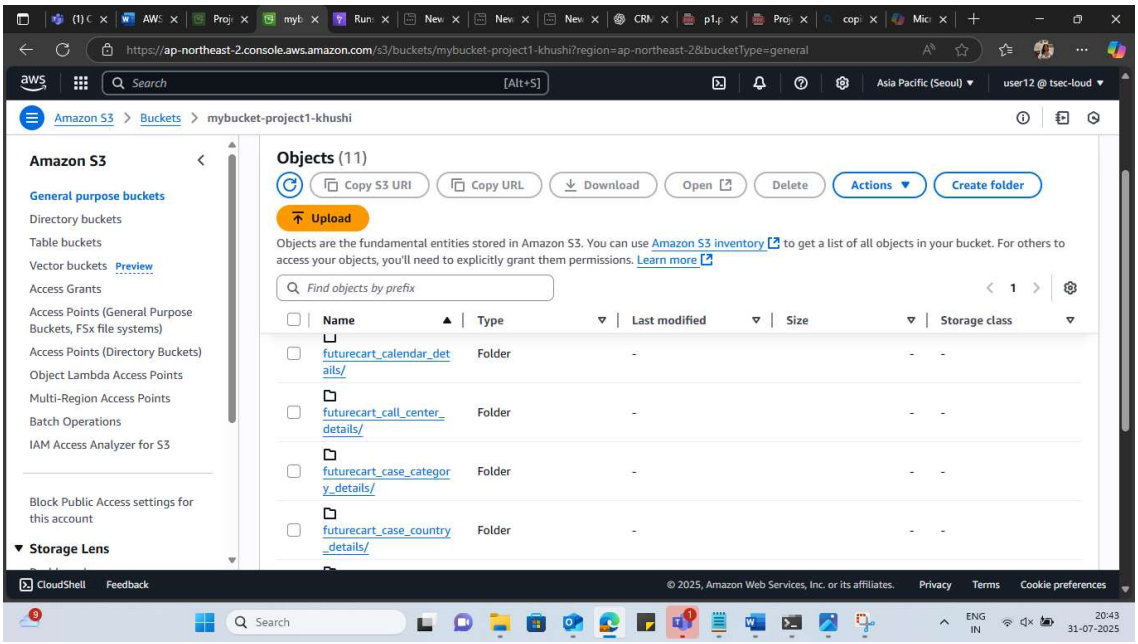
df = spark.sql("SELECT * FROM futurecart_call_center_details")

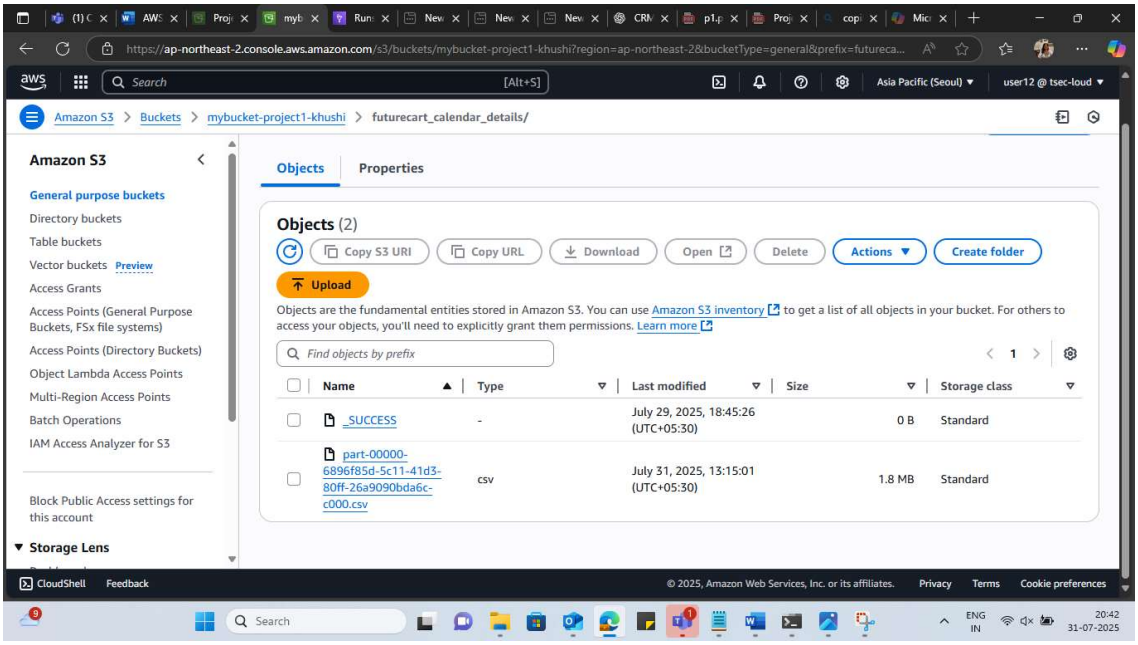
df.write.option("header", "true").csv("s3:// mybucket-project1-khushi / futurecart_call_center_details /")

- Repeated for each Hive table all rest 7 tables data.

Status

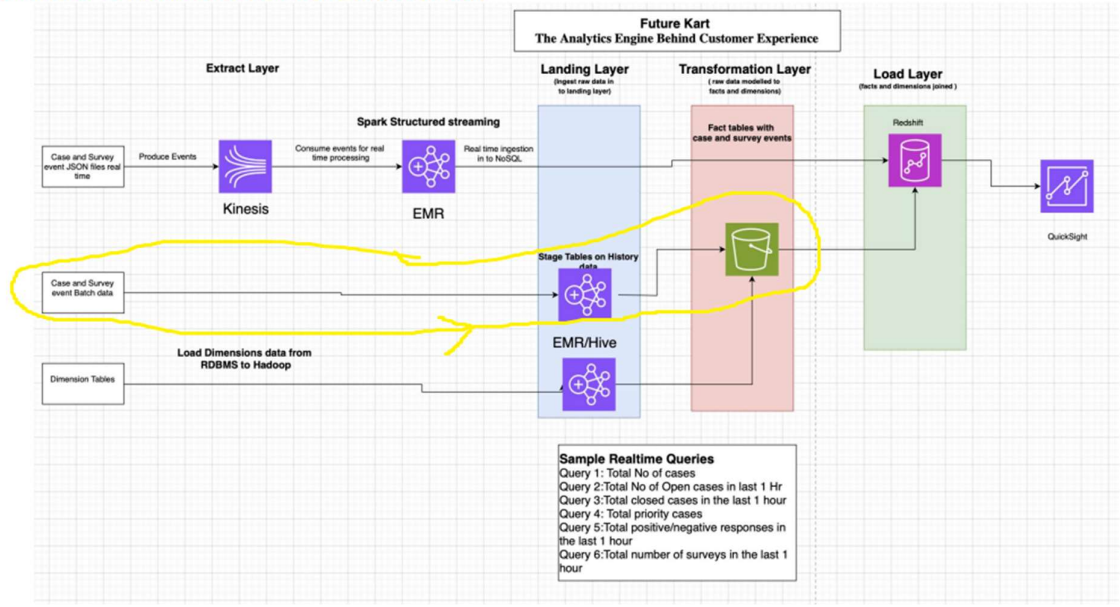
- ✅ EC2, MySQL, and S3 configured
- ✅ All dimension data uploaded from .txt → MySQL → Hive → S3
- ✅ EMR and Spark setup complete
- ✅ Hive to S3 export via Spark working with CSV output


https://ap-northeast-2.console.aws.amazon.com/s3/buckets/mybucket-project1-khushi?region=ap-northeast-2&bucketType=general
Search [Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Amazon S3 > Buckets > mybucket-project1-khushi
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets Preview
Access Grants
Access Points (General Purpose Buckets, FSx file systems)
Access Points (Directory Buckets)
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Objects (11)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
Name
Type
Last modified
Size
Storage class
futurecart_calendar_details/
Folder
futurecart_call_center_details/
Folder
futurecart_case_category_details/
Folder
futurecart_case_country_details/
Folder
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
20:43
31-07-2025


https://ap-northeast-2.console.aws.amazon.com/s3/buckets/mybucket-project1-khushi?region=ap-northeast-2&bucketType=general&prefix=futureca...
Search [Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Amazon S3 > Buckets > mybucket-project1-khushi > futurecart_calendar_details/
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets Preview
Access Grants
Access Points (General Purpose Buckets, FSx file systems)
Access Points (Directory Buckets)
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Objects
Properties
Objects (2)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
Name
Type
Last modified
Size
Storage class
_SUCCESS
-
July 29, 2025, 18:45:26 (UTC+05:30)
0 B
Standard
part-00000-6896f85d-5c11-41d3-80ff-26a9090bda6c-c000.csv
csv
July 31, 2025, 13:15:01 (UTC+05:30)
1.8 MB
Standard
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
20:42
31-07-2025

Historical Batch Data Generation & Ingestion Workflow -

3. Data Flow Architecture/Process Flow



1. Batch Data Script Uploaded to S3

- Uploaded your script generate_historical_data.py to S3.

2. Historical Data Generation

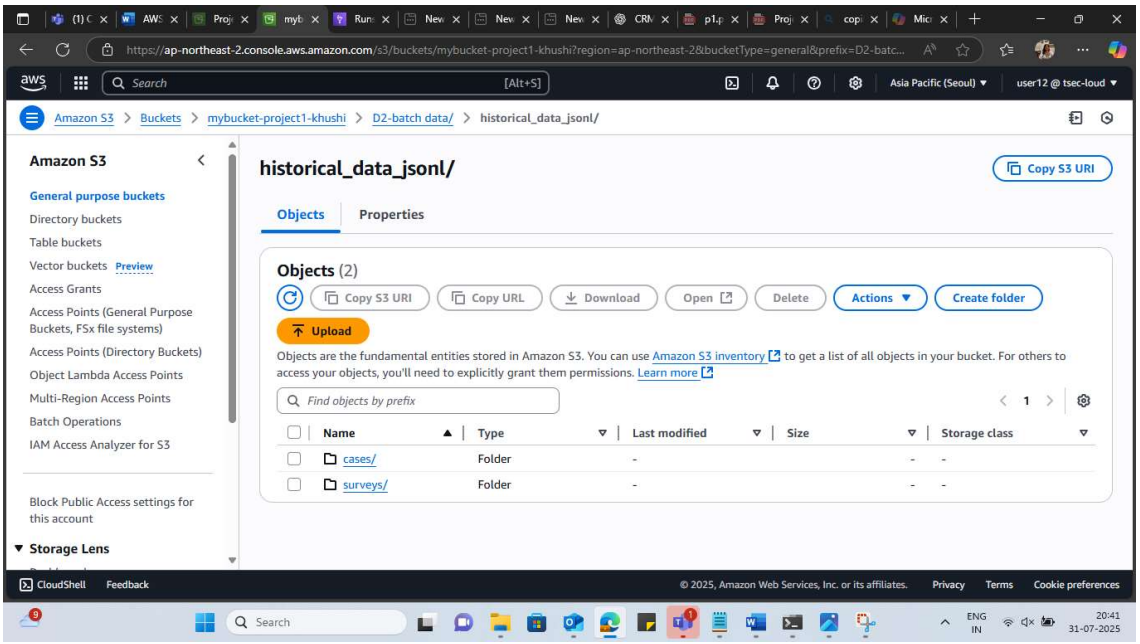
- Script generated:
    - case_data_day1.json to case_data_day10.json under historical_data_jsonl/cases/
    - survey_data_day1.json to survey_data_day10.json under historical_data_jsonl/surveys/

3. Uploaded JSON Files to S3

- Copied the generated JSON files to your S3 bucket, e.g.:

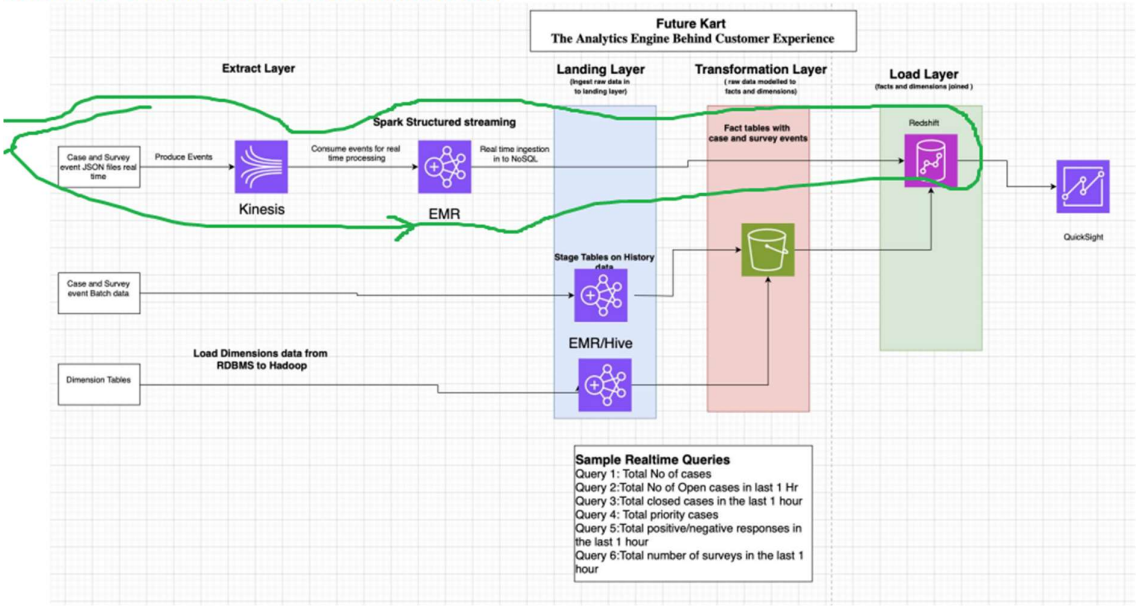
s3://mybucket-project1-khushi/historical_data_jsonl/cases/

s3://mybucket-project1-khushi/historical_data_jsonl/surveys/


https://ap-northeast-2.console.aws.amazon.com/s3/buckets/mybucket-project1-khushi?region=ap-northeast-2&bucketType=general&prefix=D2-batc...
Search
[Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Amazon S3 > Buckets > mybucket-project1-khushi > D2-batch data/ > historical_data_jsonl/
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets Preview
Access Grants
Access Points (General Purpose Buckets, FSx file systems)
Access Points (Directory Buckets)
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
historical_data_jsonl/
Copy S3 URI
Objects
Properties
Objects (2)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
Name
Type
Last modified
Size
Storage class
cases/
Folder
surveys/
Folder
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
20:41
31-07-2025

Real Time Data : Generate Real-Time JSON Events

3. Data Flow Architecture/Process Flow



Script used: stream_to_kinesis.py

nano stream_to_kinesis.py

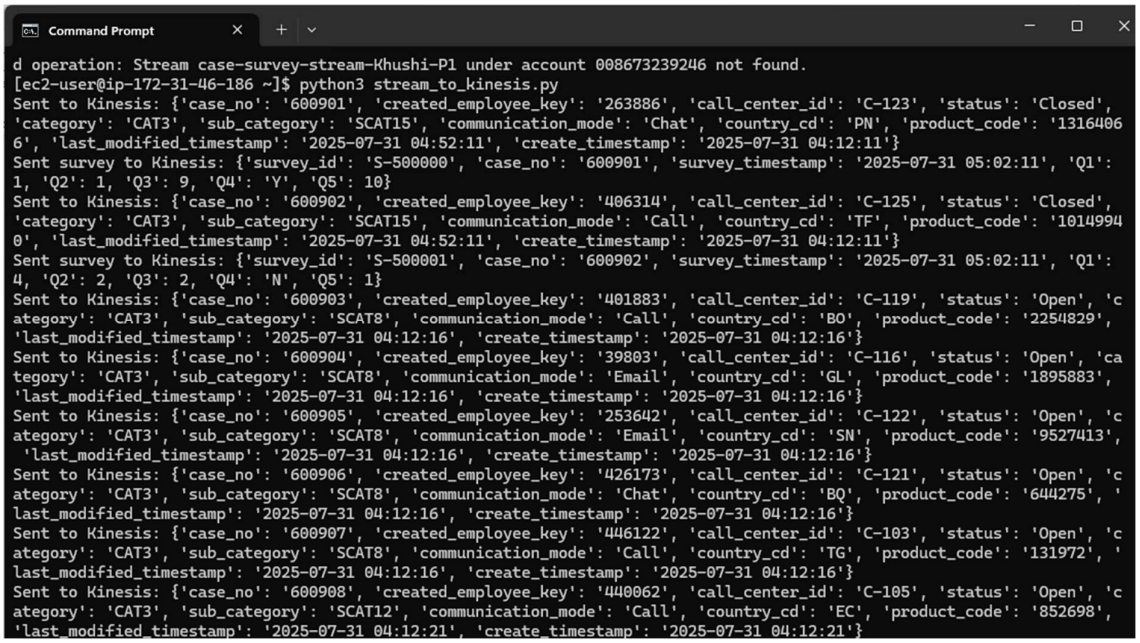
```
import boto3
import random
import time
import calendar
import json
from datetime import datetime, timedelta

# Initialize Kinesis client
kinesis = boto3.client('kinesis', region_name='ap-northeast-2')
stream_name = 'case-survey-stream-Khushi-P1'

# Read case data
with open('000000_0', 'r') as case_data_obj:
    all_case_data_lines = case_data_obj.readlines()

open_case_time_diff_mins = [40, 50, 60]
closed_case_time_diff_mins = [5, 10, 20, 30]
number_of_cases_counts = [1, 2, 3, 4, 5, 6]
scores = list(range(1, 11))
answer = ["Y", "N"]
survey_id_start = 500000
category = 'CAT3'
sub_categorys = ['SCAT8', 'SCAT9', 'SCAT10', 'SCAT11', 'SCAT12', 'SCAT13', 'SCAT14', 'SCAT15', 'SCAT16']
```

```
total_cases = len(all_case_data_lines)
i = 900

while i <= (total_cases - 1):
    sub_category = random.choice(sub_categorys)
    number_of_cases = random.choice(number_of_cases_counts)
    cases = all_case_data_lines[i:i + number_of_cases]

    current_timestamp = datetime.now()
    case_created_ts = str(current_timestamp - timedelta(minutes=random.choice(open_case_time_diff_mins)))[:19]
    case_closed_ts = str(current_timestamp - timedelta(minutes=random.choice(closed_case_time_diff_mins)))[:19]
    survey_ts = str(current_timestamp)[:19]
    file_ts = calendar.timegm(time.gmtime())

    for j in cases:
        case_no, created_employee, call_center, status, category1, sub_category1, mode, country, product =
j.strip().split(',')

        case_data = {
            "case_no": case_no,
            "created_employee_key": created_employee,
            "call_center_id": call_center,
            "status": "Closed" if i % 5 == 0 else status,
            "category": category,
            "sub_category": sub_category,
            "communication_mode": mode,
            "country_cd": country,
            "product_code": product,
            "last_modified_timestamp": case_closed_ts if i % 5 == 0 else case_created_ts,
            "create_timestamp": case_created_ts
        }

        # Send to Kinesis
        kinesis.put_record(
            StreamName=stream_name,
            Data=json.dumps(case_data),
            PartitionKey=case_no
        )
        print("Sent to Kinesis:", case_data)

        if i % 5 == 0:
            survey_data = {
                "survey_id": f"S-{survey_id_start}",
                "case_no": case_no,
                "survey_timestamp": survey_ts,
                "Q1": random.choice(scores),
                "Q2": random.choice(scores),
                "Q3": random.choice(scores),
                "Q4": random.choice(answer),
                "Q5": random.choice(scores)
            }
            kinesis.put_record(
                StreamName=stream_name,
```

```
            Data=json.dumps(survey_data),
            PartitionKey=case_no
        )
        print("Sent survey to Kinesis:", survey_data)
        survey_id_start += 1

    i += number_of_cases
    time.sleep(5)
```

pip3 install boto3

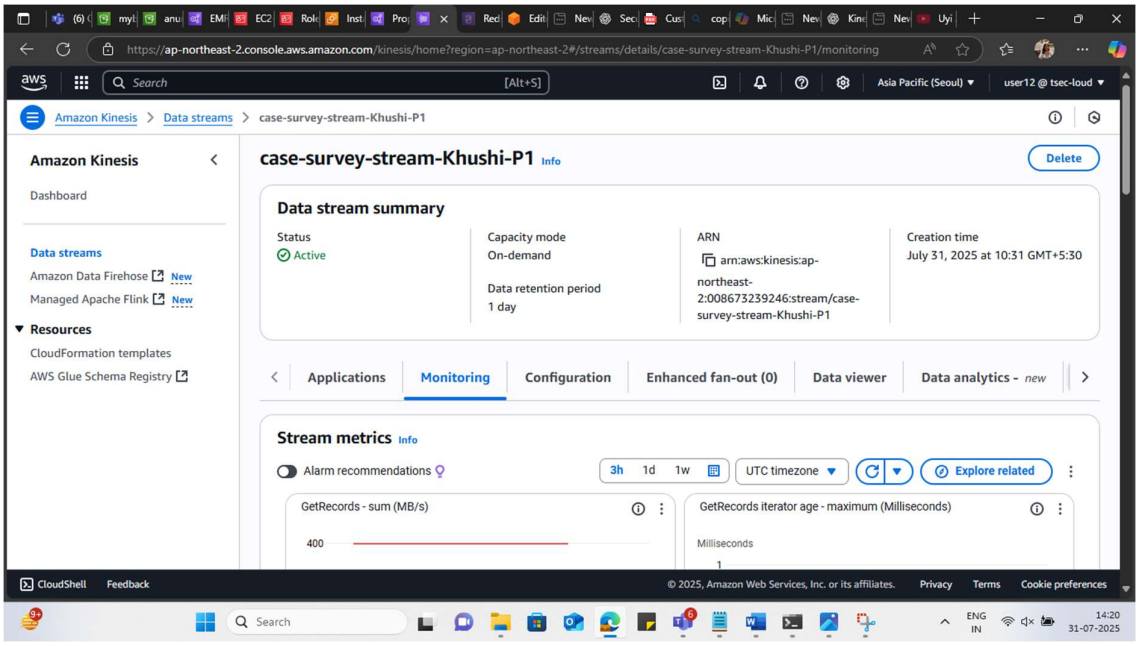
python3 stream_to_kinesis.py



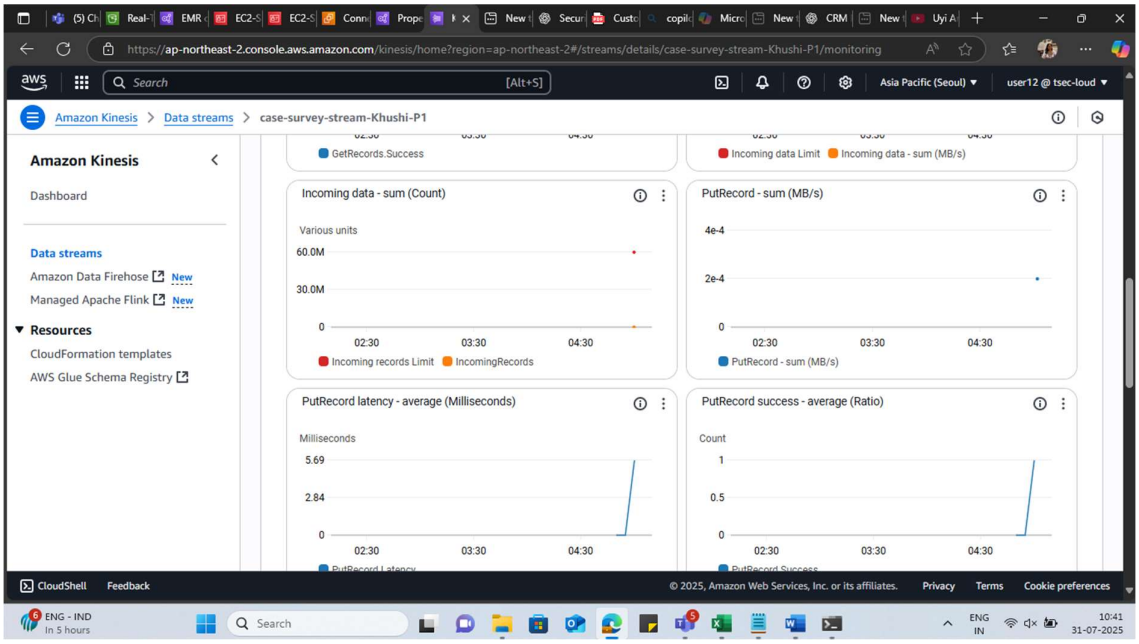
✅ Step 2: Kinesis Data Stream (Input Source for Streaming)

You created a stream named:

✅ case-survey-stream-Khushi-P1



This stream receives the real-time case/survey events from the EC2 script.



✅ Step 3: Spark Streaming Job on EMR

nano kinesis_to_redshift.py

```
#!/usr/bin/env python3
"""
FutureKart Real-time Data Pipeline - Worker-Safe Streaming Version
================================================================

This version provides streaming without UDF dependencies that cause issues
in distributed Spark environments. It uses a micro-batch approach with
direct DataFrame operations for maximum compatibility.

USAGE:
------
spark-submit \
  --packages "com.amazon.redshift:redshift-jdbc42:2.1.0.9" \
  kinesis_to_redshift_worker_safe.py

Features:
- No UDF dependencies (worker-safe)
- Uses micro-batch processing with direct Kinesis calls
- True structured streaming with writeStream
- Built-in error handling and recovery
- Works in any Spark environment

Author: FutureKart CRM Data Pipeline
Date: July 2025
"""

from pyspark.sql import SparkSession
from pyspark.sql.functions import col, from_json, to_timestamp, when, lit
from pyspark.sql.types import StructType, StructField, StringType, IntegerType
import os
import shutil
import boto3
import json
import time
import threading
from queue import Queue

def create_spark_session():
    """Create optimized Spark session for streaming"""
    return SparkSession.builder \
        .appName("FutureKart-Worker-Safe-Pipeline") \
        .config("spark.jars.packages", "com.amazon.redshift:redshift-jdbc42:2.1.0.9") \
        .config("spark.sql.adaptive.enabled", "true") \
        .config("spark.sql.adaptive.coalescePartitions.enabled", "true") \
        .config("spark.serializer", "org.apache.spark.serializer.KryoSerializer") \
        .getOrCreate()

def get_schemas():
    """Define precise schemas for case and survey events"""

    case_schema = StructType([
        StructField("case_no", StringType(), True),
        StructField("created_employee_key", StringType(), True),
        StructField("call_center_id", StringType(), True),
        StructField("status", StringType(), True),
        StructField("category", StringType(), True),
        StructField("sub_category", StringType(), True),
        StructField("communication_mode", StringType(), True),
        StructField("country_cd", StringType(), True),
        StructField("product_code", StringType(), True),
        StructField("last_modified_timestamp", StringType(), True),
```

```
        StructField("create_timestamp", StringType(), True)
    ])

    survey_schema = StructType([
        StructField("survey_id", StringType(), True),
        StructField("case_no", StringType(), True),
        StructField("survey_timestamp", StringType(), True),
        StructField("Q1", IntegerType(), True),
        StructField("Q2", IntegerType(), True),
        StructField("Q3", IntegerType(), True),
        StructField("Q4", StringType(), True),
        StructField("Q5", IntegerType(), True)
    ])

    return case_schema, survey_schema

class KinesisDataCollector:
    """Thread-safe Kinesis data collector"""

    def __init__(self, stream_name, region):
        self.stream_name = stream_name
        self.region = region
        self.kinesis_client = boto3.client('kinesis', region_name=region)
        self.data_queue = Queue()
        self.running = False
        self.shard_iterators = {}

    def start_collection(self):
        """Start collecting data from Kinesis"""
        self.running = True
        thread = threading.Thread(target=self._collect_data)
        thread.daemon = True
        thread.start()
        return thread

    def stop_collection(self):
        """Stop collecting data"""
        self.running = False

    def get_batch_data(self):
        """Get a batch of collected data"""
        data = []
        while not self.data_queue.empty() and len(data) < 100:
            data.append(self.data_queue.get())
        return data

    def _collect_data(self):
        """Background thread to collect data from Kinesis"""
        print("🔄 Starting Kinesis data collection thread...")

        while self.running:
            try:
                # Get stream description
                stream_desc = self.kinesis_client.describe_stream(StreamName=self.stream_name)
                shards = stream_desc['StreamDescription']['Shards']

                for shard in shards[:1]:  # Process first shard for now
                    shard_id = shard['ShardId']

                    # Get or create shard iterator
                    if shard_id not in self.shard_iterators:
```

```
                response = self.kinesis_client.get_shard_iterator(
                    StreamName=self.stream_name,
                    ShardId=shard_id,
                    ShardIteratorType='LATEST'
                )
                self.shard_iterators[shard_id] = response['ShardIterator']

            # Get records
            try:
                response = self.kinesis_client.get_records(
                    ShardIterator=self.shard_iterators[shard_id],
                    Limit=50
                )

                records = response.get('Records', [])
                if records:
                    print(f" 📦 Collected {len(records)} records from shard {shard_id}")

                for record in records:
                    try:
                        data = record['Data']
                        if isinstance(data, bytes):
                            data = data.decode('utf-8')

                        record_data = {
                            "json_data": str(data),
                            "partitionKey": record.get('PartitionKey', ''),
                            "sequenceNumber": record.get('SequenceNumber', ''),
                            "approximateArrivalTimestamp": str(record.get('ApproximateArrivalTimestamp', ''))
                        }
                        self.data_queue.put(record_data)

                    except Exception as e:
                        print(f" ⚠️ Error processing record: {str(e)}")

                # Update shard iterator
                if response.get('NextShardIterator'):
                    self.shard_iterators[shard_id] = response['NextShardIterator']
                else:
                    # Iterator expired, get a new one
                    response = self.kinesis_client.get_shard_iterator(
                        StreamName=self.stream_name,
                        ShardId=shard_id,
                        ShardIteratorType='LATEST'
                    )
                    self.shard_iterators[shard_id] = response['ShardIterator']

            except Exception as e:
                print(f" ⚠️ Error reading from shard {shard_id}: {str(e)}")

        # Wait before next collection
        time.sleep(5)

    except Exception as e:
        print(f" ⚠️ Error in data collection: {str(e)}")
        time.sleep(10)

def create_rate_stream(spark):
    """Create a simple rate stream for triggering"""
    return spark \
        .readStream \
```

```
        .format("rate") \
        .option("rowsPerSecond", 1) \
        .option("numPartitions", 1) \
        .load()

def process_kinesis_data(records_data, case_schema, survey_schema):
    """Process Kinesis records and separate into case and survey events"""
    case_events = []
    survey_events = []

    for record in records_data:
        json_data = record['json_data']

        # Event type detection
        if any(field in json_data for field in ['survey_id', 'survey_timestamp', '"Q1":', '"Q2":']):
            # Survey event
            try:
                event_data = json.loads(json_data)
                survey_events.append(event_data)
            except:
                print(f"⚠️ Invalid survey JSON: {json_data[:50]}...")
        elif any(field in json_data for field in ['call_center_id', 'communication_mode', 'created_employee_key', '"status":']):
            # Case event
            try:
                event_data = json.loads(json_data)
                case_events.append(event_data)
            except:
                print(f"⚠️ Invalid case JSON: {json_data[:50]}...")

    return case_events, survey_events

def write_to_redshift_batch(spark, events, table_name, schema, redshift_config):
    """Write a batch of events to Redshift"""
    if not events:
        return

    try:
        # Create DataFrame from events
        df = spark.createDataFrame(events, schema)

        # Transform timestamps for case events
        if table_name == "futurecart_case_details":
            df = df.select(
                col("case_no"),
                col("created_employee_key"),
                col("call_center_id"),
                col("status"),
                col("category"),
                col("sub_category"),
                col("communication_mode"),
                col("country_cd"),
                col("product_code"),
                to_timestamp(col("last_modified_timestamp"), "yyyy-MM-dd HH:mm:ss").alias("last_modified_timestamp"),
                to_timestamp(col("create_timestamp"), "yyyy-MM-dd HH:mm:ss").alias("create_timestamp")
            )

        # Transform timestamps for survey events
        elif table_name == "futurecart_case_survey_details":
            df = df.select(
                col("survey_id"),
                col("case_no"),
```

```
            to_timestamp(col("survey_timestamp"), "yyyy-MM-dd HH:mm:ss").alias("survey_timestamp"),
            col("Q1"),
            col("Q2"),
            col("Q3"),
            col("Q4"),
            col("Q5")
        )

        # Write to Redshift
        df.write \
            .format("jdbc") \
            .option("url", redshift_config["url"]) \
            .option("dbtable", table_name) \
            .option("user", redshift_config["user"]) \
            .option("password", redshift_config["password"]) \
            .option("driver", redshift_config["driver"]) \
            .mode("append") \
            .save()

        print(f"✅ Successfully wrote {len(events)} records to {table_name}")

    except Exception as e:
        print(f"❌ Error writing to {table_name}: {str(e)}")

def test_redshift_connection(spark, redshift_config):
    """Test Redshift connection"""
    print("🔍 Testing Redshift connection...")

    try:
        test_df = spark.read \
            .format("jdbc") \
            .option("url", redshift_config["url"]) \
            .option("dbtable", "(SELECT 1 as test_col) as test_query") \
            .option("user", redshift_config["user"]) \
            .option("password", redshift_config["password"]) \
            .option("driver", redshift_config["driver"]) \
            .load()

        result = test_df.collect()
        print(f"✅ Redshift connection successful!")
        return True

    except Exception as e:
        print(f"❌ Redshift connection failed: {str(e)}")
        return False

def main():
    """Main streaming pipeline"""

    print("🚀 Starting FutureKart Worker-Safe Streaming Pipeline...")
    print("=" * 60)

    # Initialize Spark
    spark = create_spark_session()
    spark.sparkContext.setLogLevel("WARN")

    # Get schemas
    case_schema, survey_schema = get_schemas()

    # Redshift configuration
    redshift_config = {
```

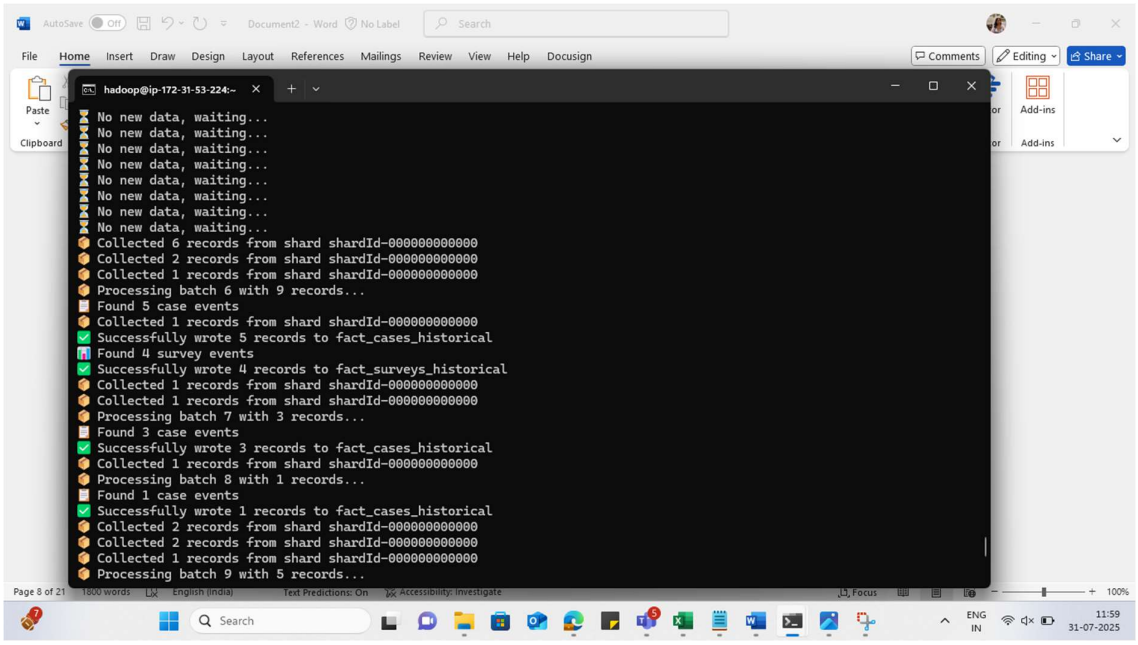
```
    "url": "jdbc:redshift://futurecart-wg-khushi.008673239246.ap-northeast-2.redshift-serverless.amazonaws.com:5439/dev",
    "user": "admin",
    "password": "Khushi12345",
    "driver": "com.amazon.redshift.jdbc.Driver"
}

print(f"🔗 Redshift Connection: {redshift_config['url']}")
print(f"👤 User: {redshift_config['user']}")

# Test Redshift connection
if not test_redshift_connection(spark, redshift_config):
    print("❌ Redshift connection test failed. Exiting...")
    return

# Initialize Kinesis collector
collector = KinesisDataCollector("case-survey-stream-Khushi-P1", "ap-northeast-2")
collector_thread = collector.start_collection()

print("=" * 60)
print("🎯 WORKER-SAFE STREAMING PIPELINE STATUS")
print("=" * 60)
print("✅ Kinesis Stream: sahil-proj1-futurekart-stream")
print("✅ Method: Background thread collection + micro-batch processing")
print("✅ Case Events → fact_cases_historical")
print("✅ Survey Events → fact_surveys_historical")
print("✅ Processing Interval: 15 seconds")
print("✅ Worker Safety: No UDF dependencies")
print("=" * 60)
print("🔄 Pipeline running... Press Ctrl+C to stop")
print("=" * 60)

try:
    batch_count = 0
    while True:
        # Get batch data from collector
        batch_data = collector.get_batch_data()

        if batch_data:
            print(f"📦 Processing batch {batch_count} with {len(batch_data)} records...")

            # Process the data
            case_events, survey_events = process_kinesis_data(batch_data, case_schema, survey_schema)

            # Write to Redshift
            if case_events:
                print(f"📋 Found {len(case_events)} case events")
                write_to_redshift_batch(spark, case_events, "fact_cases_historical", case_schema, redshift_config)

            if survey_events:
                print(f"📊 Found {len(survey_events)} survey events")
                write_to_redshift_batch(spark, survey_events, "fact_surveys_historical", survey_schema, redshift_config)

            if not case_events and not survey_events:
                print(f"⚠️ No valid events found in batch")

            batch_count += 1
        else:
            print("⏳ No new data, waiting...")

        # Wait before next batch
```

```
        time.sleep(15)

    except KeyboardInterrupt:
        print("\n🔴 Pipeline stopped by user")
    except Exception as e:
        print(f"❌ Pipeline error: {str(e)}")
        import traceback
        traceback.print_exc()
    finally:
        print("🔄 Stopping Kinesis collector...")
        collector.stop_collection()
        collector_thread.join(timeout=5)

        print("🔄 Cleaning up Spark session...")
        spark.stop()
        print("✅ Pipeline shutdown complete")

if __name__ == "__main__":
    main()
```

pip3 install boto3
sudo su -hadoop

spark-submit --packages "com.amazon.redshift:redshift-jdbc42:2.1.0.9" kinesis_to_redshift.py

```
25/07/31 05:59:40 INFO YarnClientSchedulerBackend: SchedulerBackend is ready for scheduling beginning after reached minR
egisteredResourcesRatio: 0.0
🔗 Redshift Connection: jdbc:redshift://futurecart-wg-khushi.008673239246.ap-northeast-2.redshift-serverless.amazonaws.c
om:5439/dev
👤 User: admin
🔍 Testing Redshift connection...
✅ Redshift connection successful!
/home/hadoop/.local/lib/python3.7/site-packages/boto3/compat.py:82: PythonDeprecationWarning: Boto3 will no longer suppo
rt Python 3.7 starting December 13, 2023. To continue receiving service updates, bug fixes, and security updates please
upgrade to Python 3.8 or later. More information can be found here: https://aws.amazon.com/blogs/developer/python-suppor
t-policy-updates-for-aws-sdks-and-tools/
  warnings.warn(warning, PythonDeprecationWarning)
🔄 Starting Kinesis data collection thread...============================================================
🎯 WORKER-SAFE STREAMING PIPELINE STATUS
============================================================
✅ Kinesis Stream: sahil-proj1-futurekart-stream
✅ Method: Background thread collection + micro-batch processing
✅ Case Events → fact_cases_historical
✅ Survey Events → fact_surveys_historical
✅ Processing Interval: 15 seconds
✅ Worker Safety: No UDF dependencies
============================================================
🔄 Pipeline running... Press Ctrl+C to stop
============================================================
⏳ No new data, waiting...
```



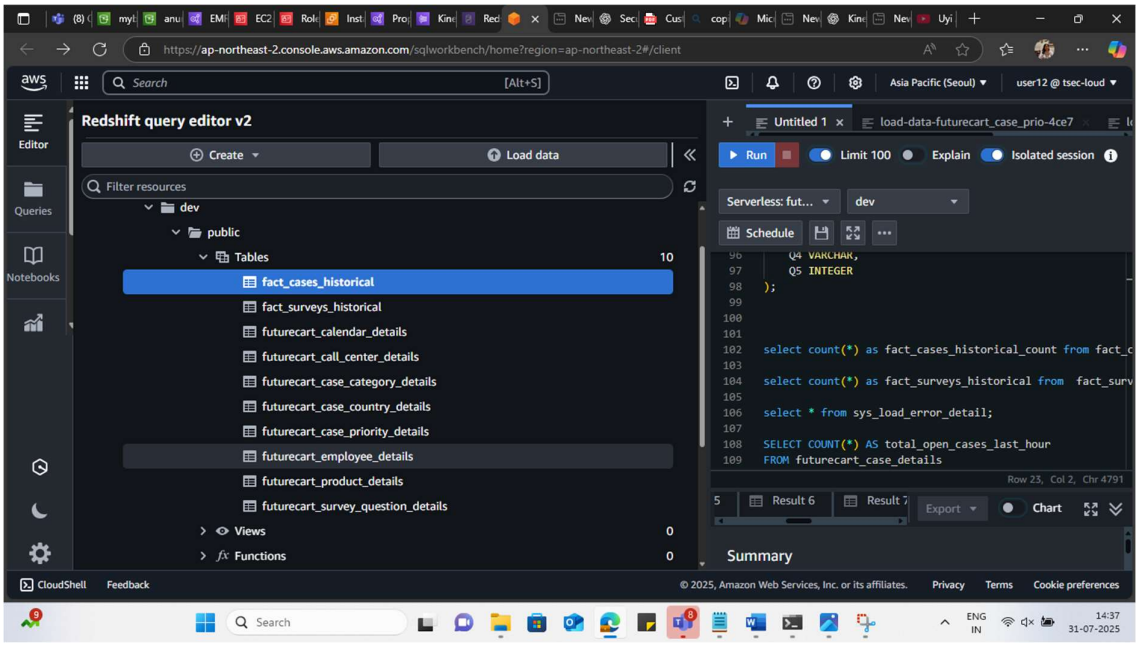
✅ Step 4: Amazon Redshift (Final Destination)

✅ 1. Created Redshift Tables

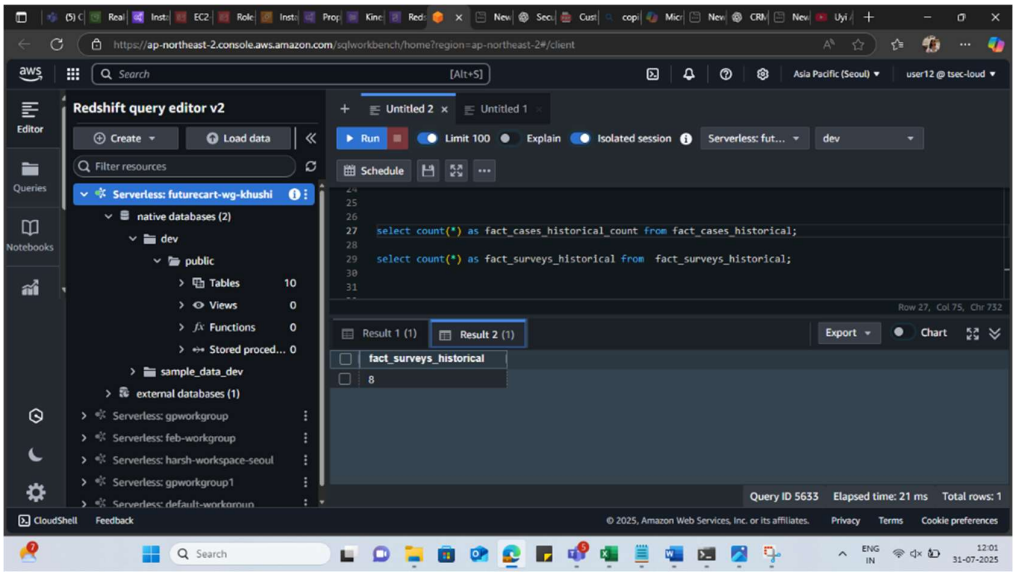
You used SQL DDL (Data Definition Language) to create dimension and fact tables in your Redshift Serverless workgroup futurecart-wg-khushi.
These include:

- 🔹 Dimension Tables:
  - futurecart_calendar_details
  - futurecart_call_center_details
  - futurecart_case_category_details
  - futurecart_case_country_details
  - futurecart_case_priority_details
  - futurecart_employee_details
  - futurecart_product_details
  - futurecart_survey_question_details
- 🔹 Fact Tables:
  - futurecart_case_details — holds case data

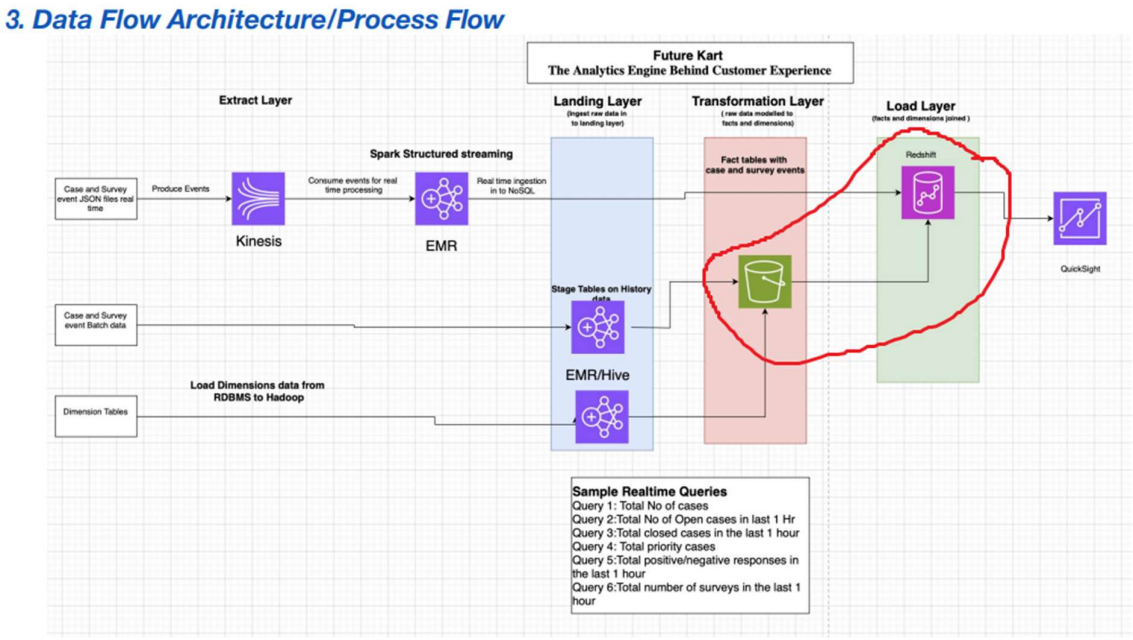
- futurecart_case_survey_details — holds survey data



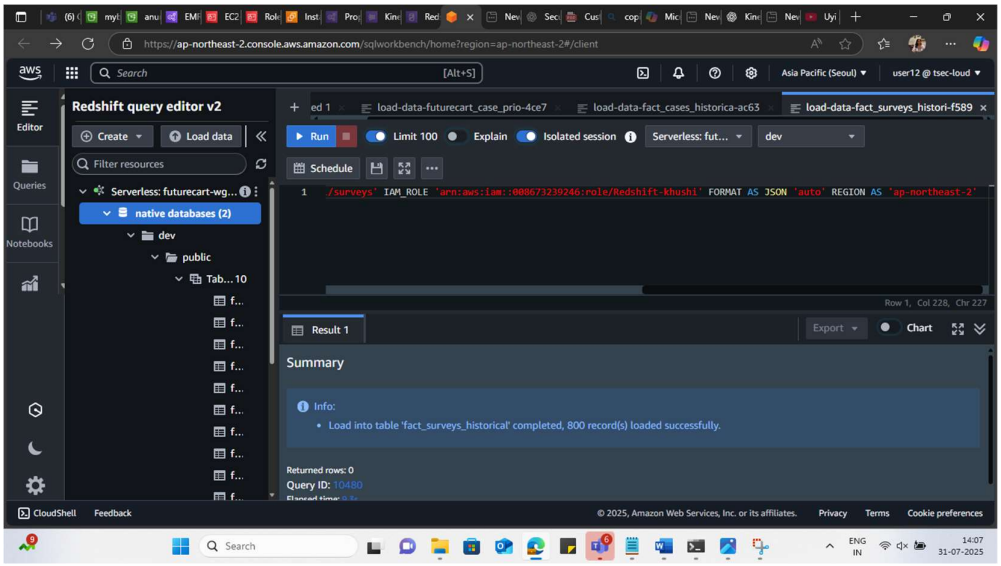
Checked getting the data or not:



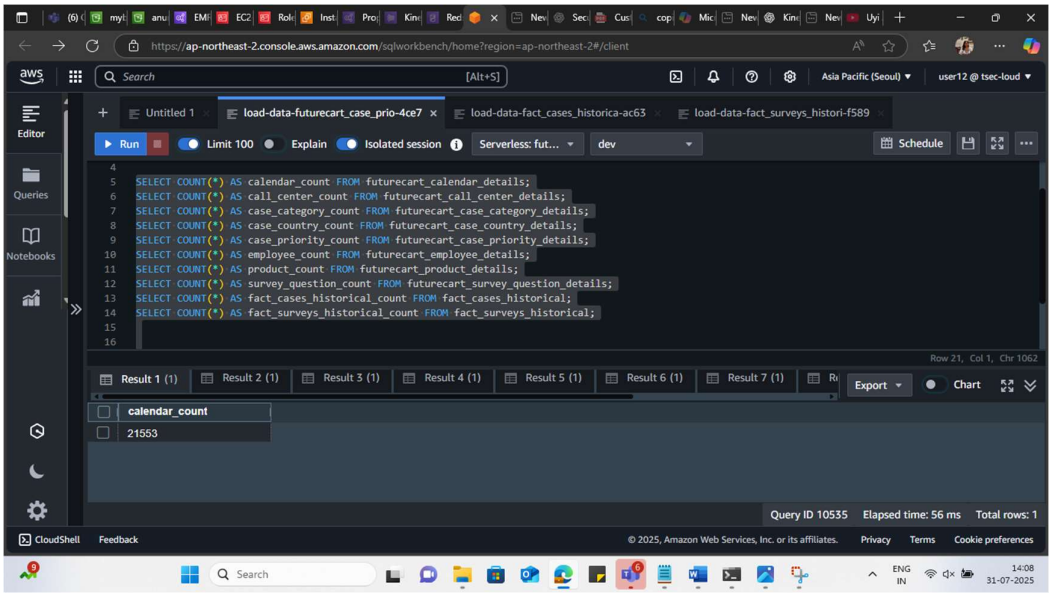
Now Uploading all the data present in s3 to Redshift tables:



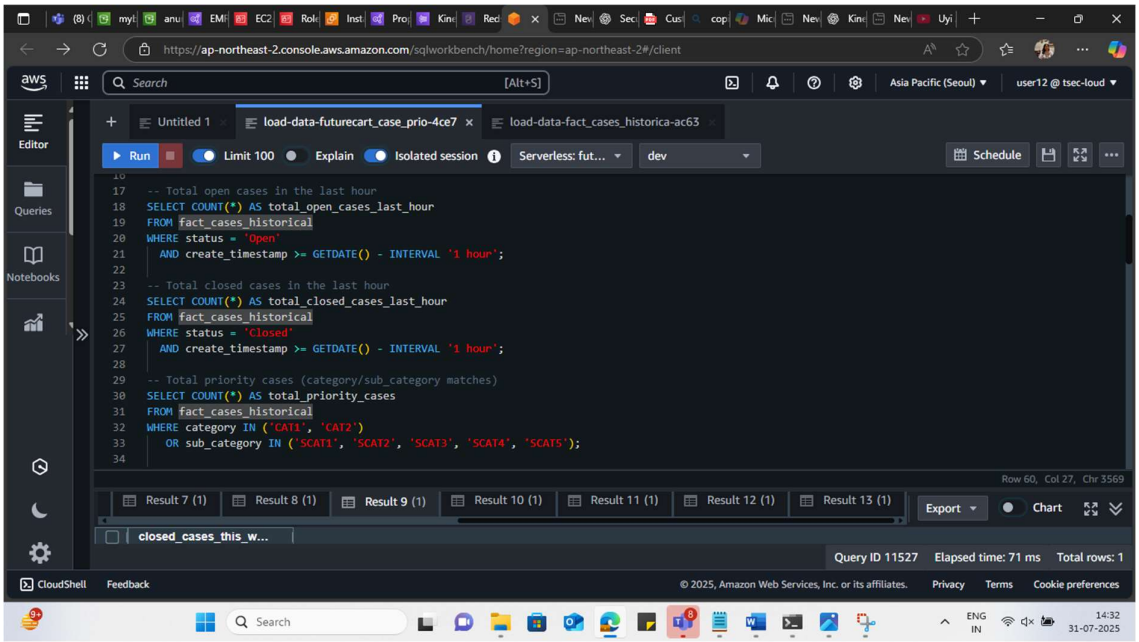
Then loaded all my data from s3 bucket to Redshift Dimension tables:



Checked if the data loaded or not:



Executed Queries:



```sql
-- Total open cases in the last hour
SELECT COUNT(*) AS total_open_cases_last_hour
FROM fact_cases_historical
WHERE status = 'Open'
  AND create_timestamp >= GETDATE() - INTERVAL '1 hour';
```

```sql
-- Total closed cases in the last hour
SELECT COUNT(*) AS total_closed_cases_last_hour
FROM fact_cases_historical
WHERE status = 'Closed'
  AND create_timestamp >= GETDATE() - INTERVAL '1 hour';
```

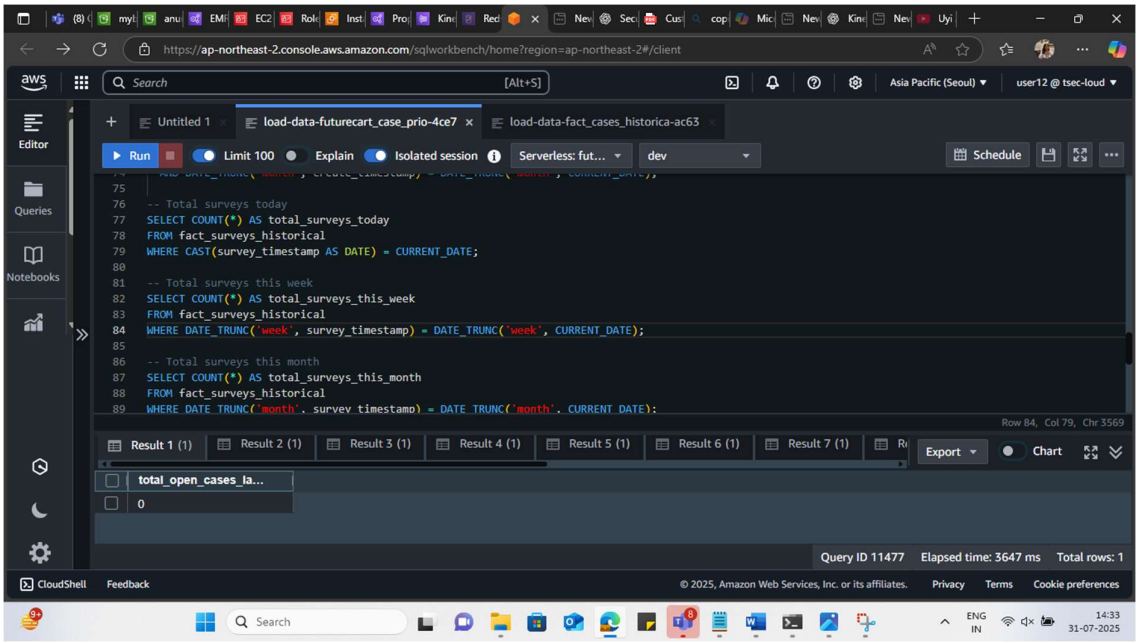
```sql
-- Total priority cases (category/sub_category matches)
SELECT COUNT(*) AS total_priority_cases
FROM fact_cases_historical
WHERE category IN ('CAT1', 'CAT2')
   OR sub_category IN ('SCAT1', 'SCAT2', 'SCAT3', 'SCAT4', 'SCAT5');
```

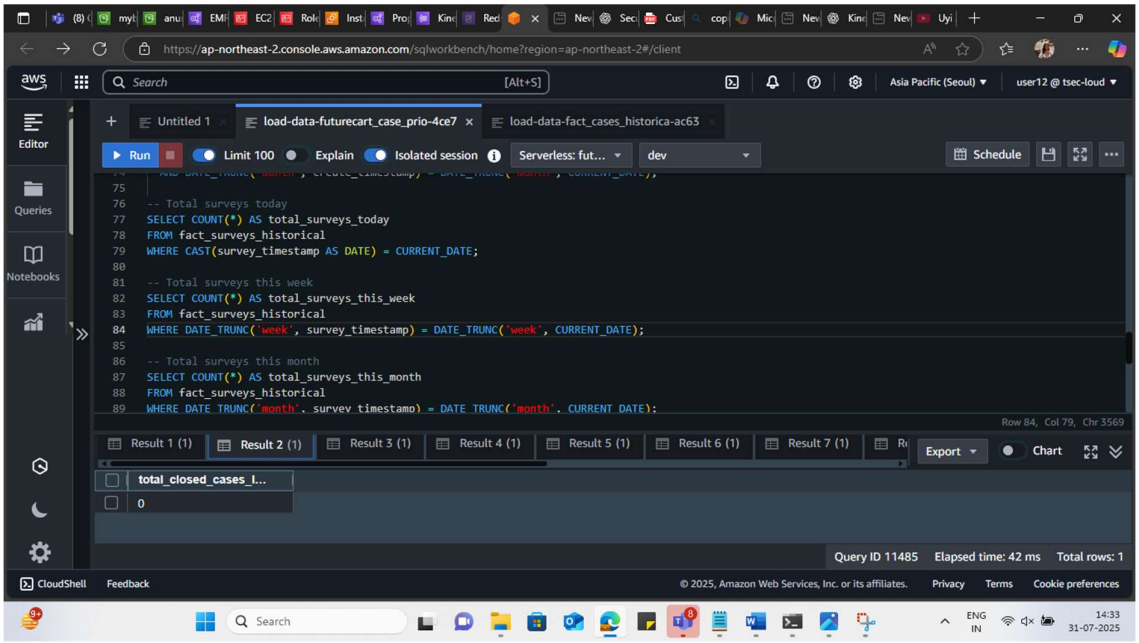
```sql
-- Total surveys in the last hour
SELECT COUNT(*) AS total_surveys_last_hour
FROM fact_surveys_historical
WHERE survey_timestamp >= GETDATE() - INTERVAL '1 hour';
```

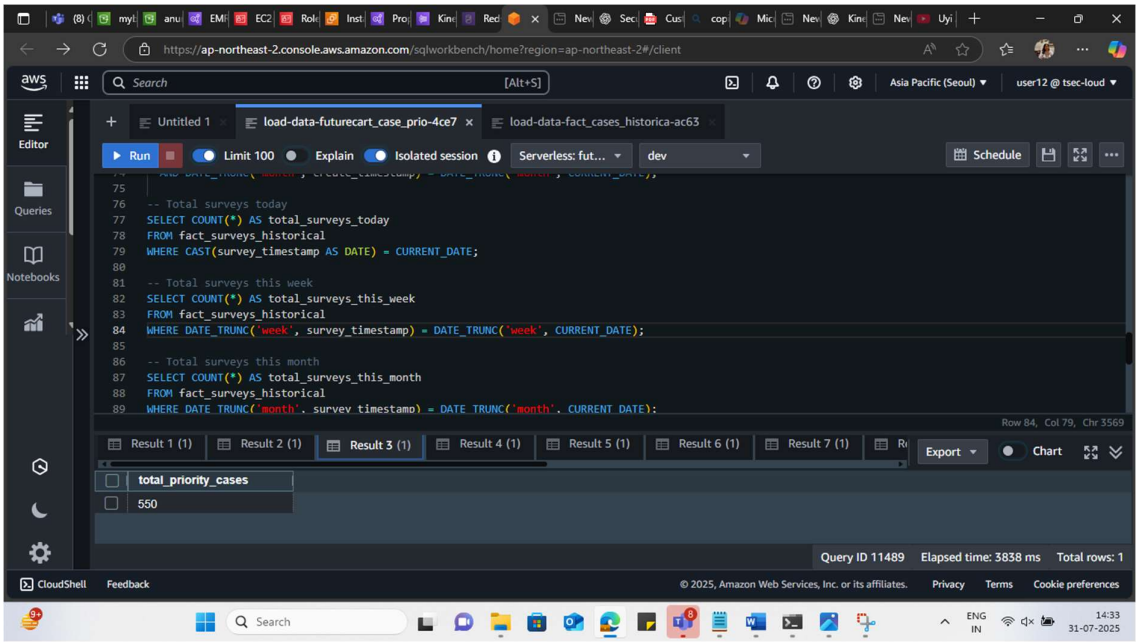
```sql
-- Open cases today
SELECT COUNT(*) AS open_cases_today
FROM fact_cases_historical
WHERE status = 'Open'
  AND CAST(create_timestamp AS DATE) = CURRENT_DATE;
```

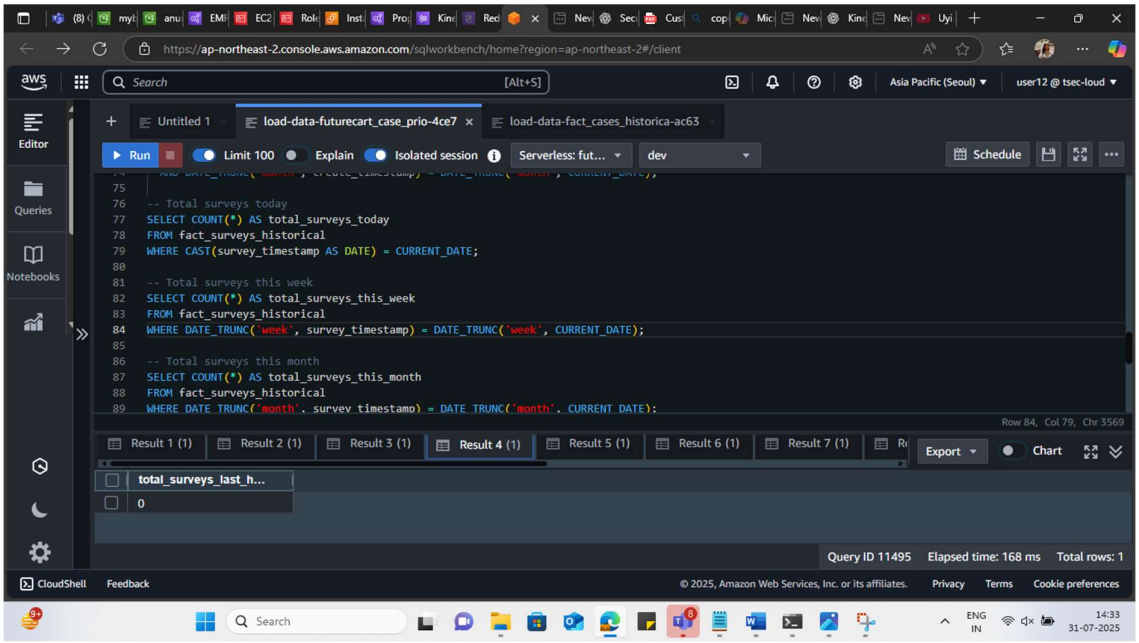
```sql
-- Open cases this week
SELECT COUNT(*) AS open_cases_this_week
FROM fact_cases_historical
WHERE status = 'Open'
  AND DATE_TRUNC('week', create_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
```

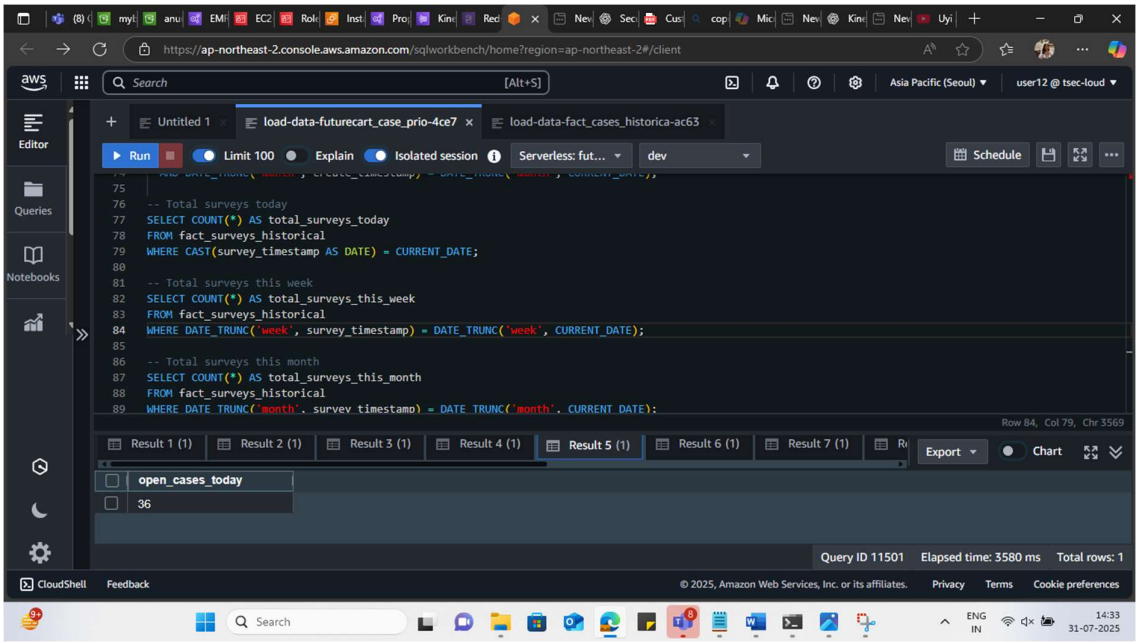
```sql
-- Open cases this month
SELECT COUNT(*) AS open_cases_this_month
FROM fact_cases_historical
WHERE status = 'Open'
  AND DATE_TRUNC('month', create_timestamp) = DATE_TRUNC('month', CURRENT_DATE);

-- Closed cases today
SELECT COUNT(*) AS closed_cases_today
FROM fact_cases_historical
WHERE status = 'Closed'
  AND CAST(create_timestamp AS DATE) = CURRENT_DATE;

-- Closed cases this week
SELECT COUNT(*) AS closed_cases_this_week
FROM fact_cases_historical
WHERE status = 'Closed'
  AND DATE_TRUNC('week', create_timestamp) = DATE_TRUNC('week', CURRENT_DATE);

-- Closed cases this month
SELECT COUNT(*) AS closed_cases_this_month
FROM fact_cases_historical
WHERE status = 'Closed'
  AND DATE_TRUNC('month', create_timestamp) = DATE_TRUNC('month', CURRENT_DATE);

-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;

-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);

-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
```

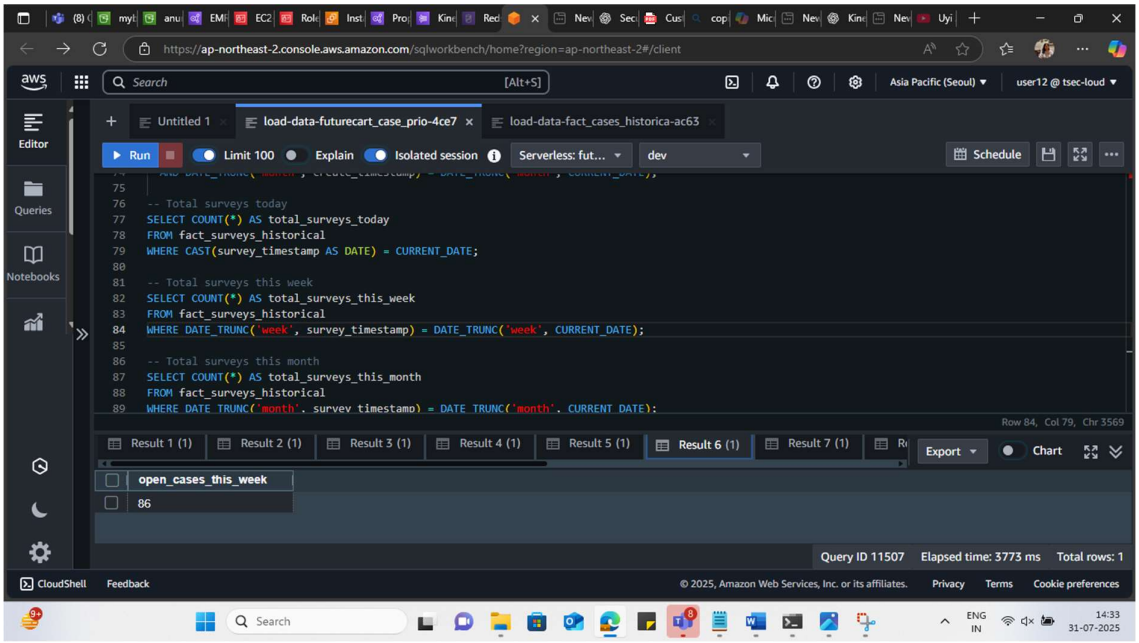

https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search
[Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Editor
Queries
Notebooks
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 1 (1)
Result 2 (1)
Result 3 (1)
Result 4 (1)
Result 5 (1)
Result 6 (1)
Result 7 (1)
Export
Chart
total_open_cases_la...
0
Query ID 11477
Elapsed time: 3647 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:33
31-07-2025

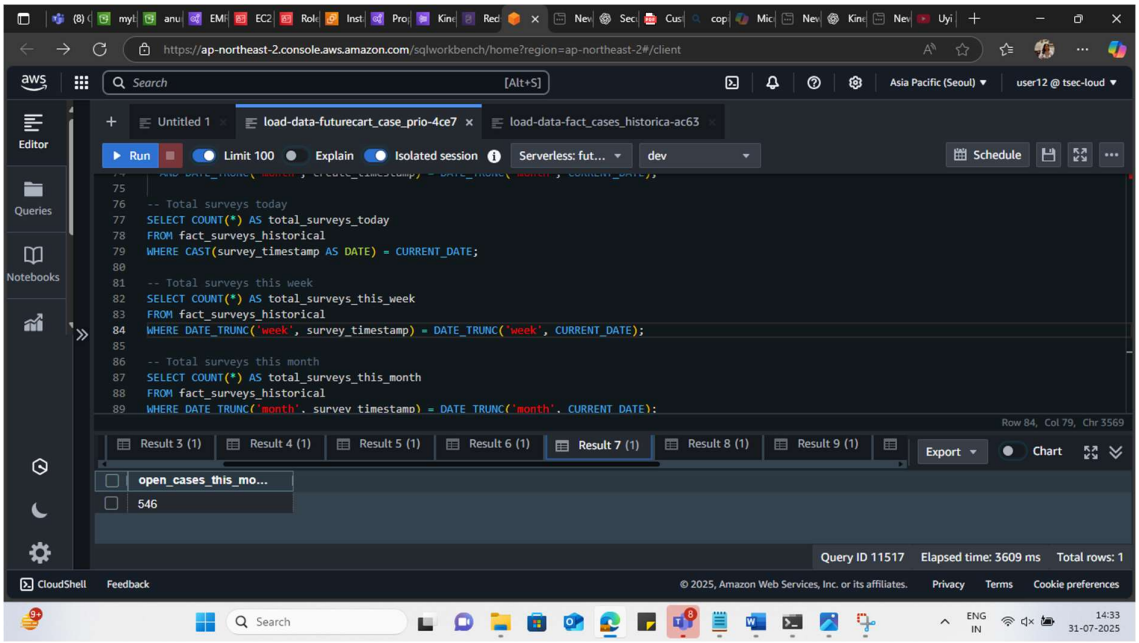

https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search
[Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Editor
Queries
Notebooks
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 1 (1)
Result 2 (1)
Result 3 (1)
Result 4 (1)
Result 5 (1)
Result 6 (1)
Result 7 (1)
Export
Chart
total_closed_cases_l...
0
Query ID 11485
Elapsed time: 42 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:33
31-07-2025

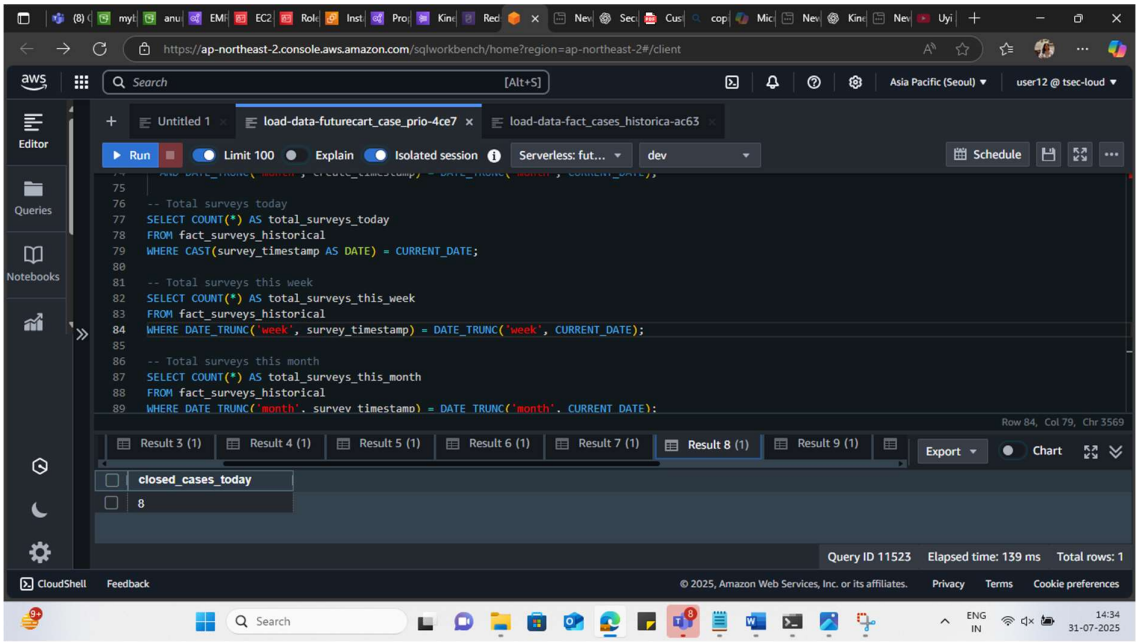

https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search [Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
Editor
Queries
Notebooks
-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 1 (1)
Result 2 (1)
Result 3 (1)
Result 4 (1)
Result 5 (1)
Result 6 (1)
Result 7 (1)
Export
Chart
total_priority_cases
550
Query ID 11489
Elapsed time: 3838 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:33
31-07-2025

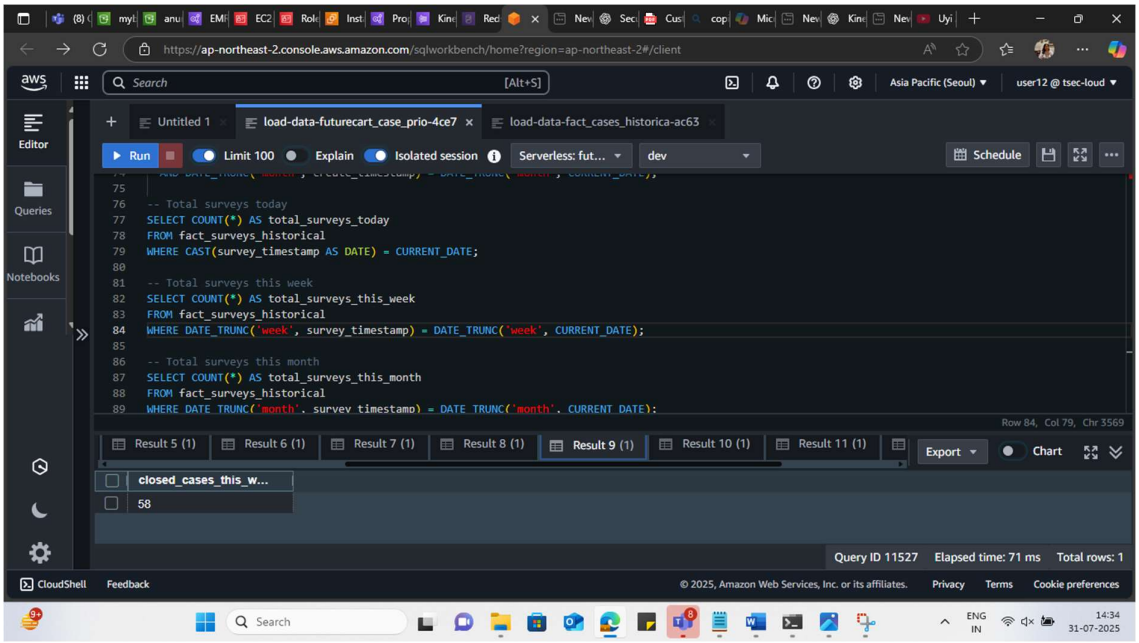

https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search [Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
Editor
Queries
Notebooks
-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 1 (1)
Result 2 (1)
Result 3 (1)
Result 4 (1)
Result 5 (1)
Result 6 (1)
Result 7 (1)
Export
Chart
total_surveys_last_h...
0
Query ID 11495
Elapsed time: 168 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:33
31-07-2025

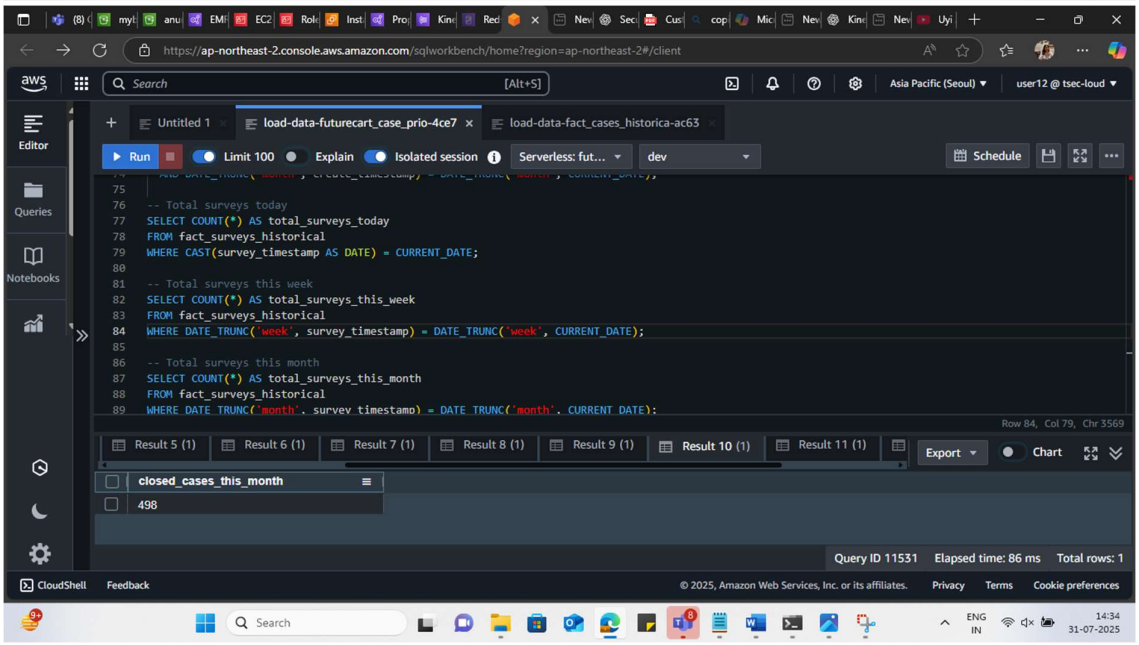

https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search
[Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
Editor
Queries
Notebooks
-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 1 (1)
Result 2 (1)
Result 3 (1)
Result 4 (1)
Result 5 (1)
Result 6 (1)
Result 7 (1)
Export
Chart
open_cases_today
36
Query ID 11501
Elapsed time: 3580 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:33
31-07-2025

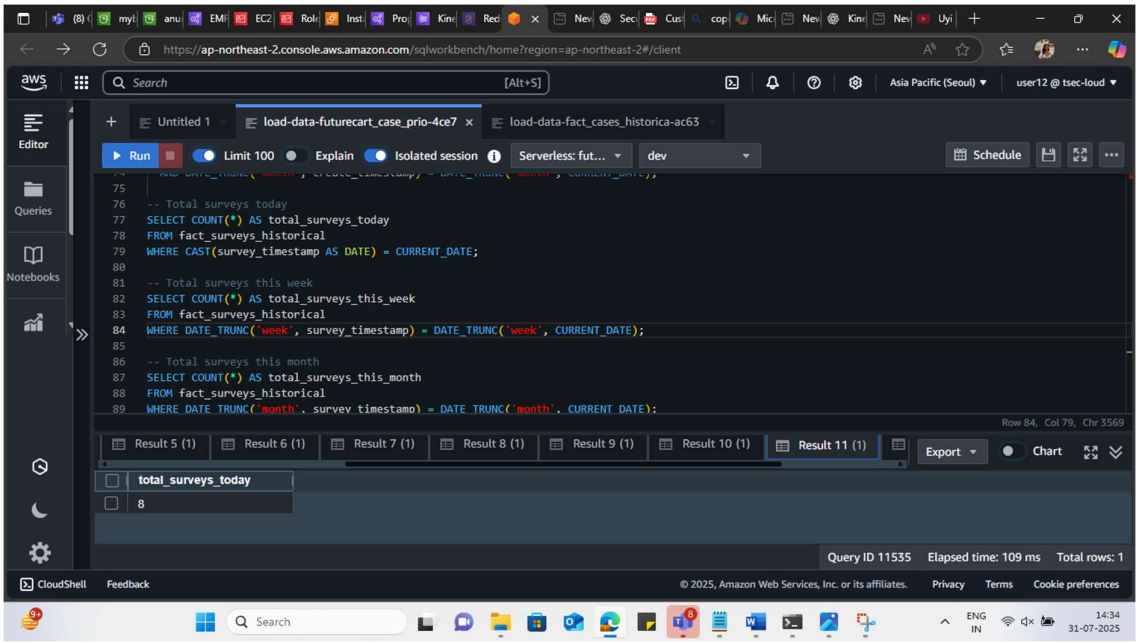

https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search
[Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
Editor
Queries
Notebooks
-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 1 (1)
Result 2 (1)
Result 3 (1)
Result 4 (1)
Result 5 (1)
Result 6 (1)
Result 7 (1)
Export
Chart
open_cases_this_week
86
Query ID 11507
Elapsed time: 3773 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:33
31-07-2025

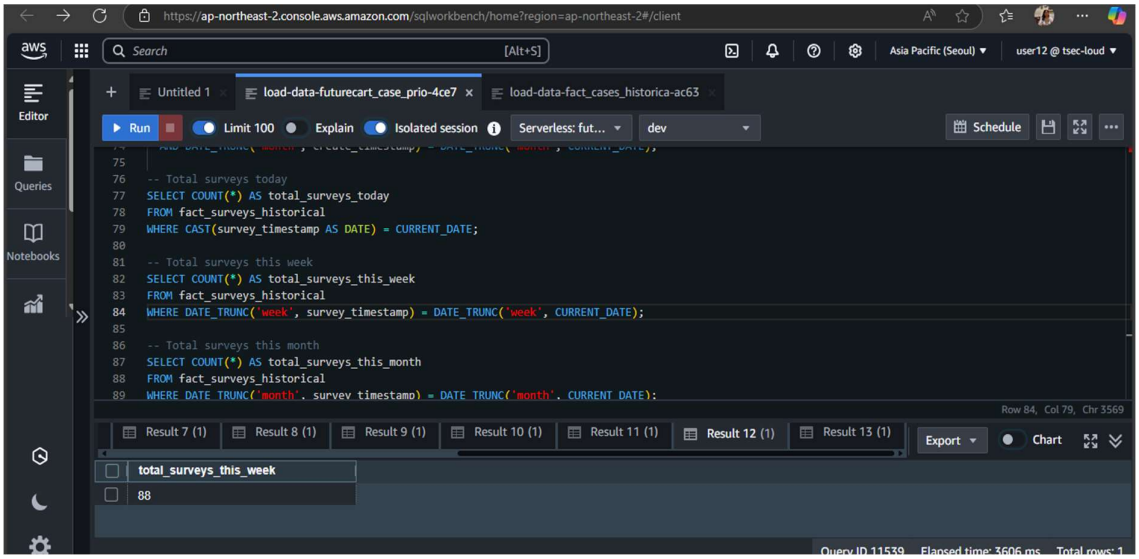

https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search [Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
Editor
Queries
Notebooks
-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 3 (1)
Result 4 (1)
Result 5 (1)
Result 6 (1)
Result 7 (1)
Result 8 (1)
Result 9 (1)
Export
Chart
open_cases_this_mo...
546
Query ID 11517
Elapsed time: 3609 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:33
31-07-2025

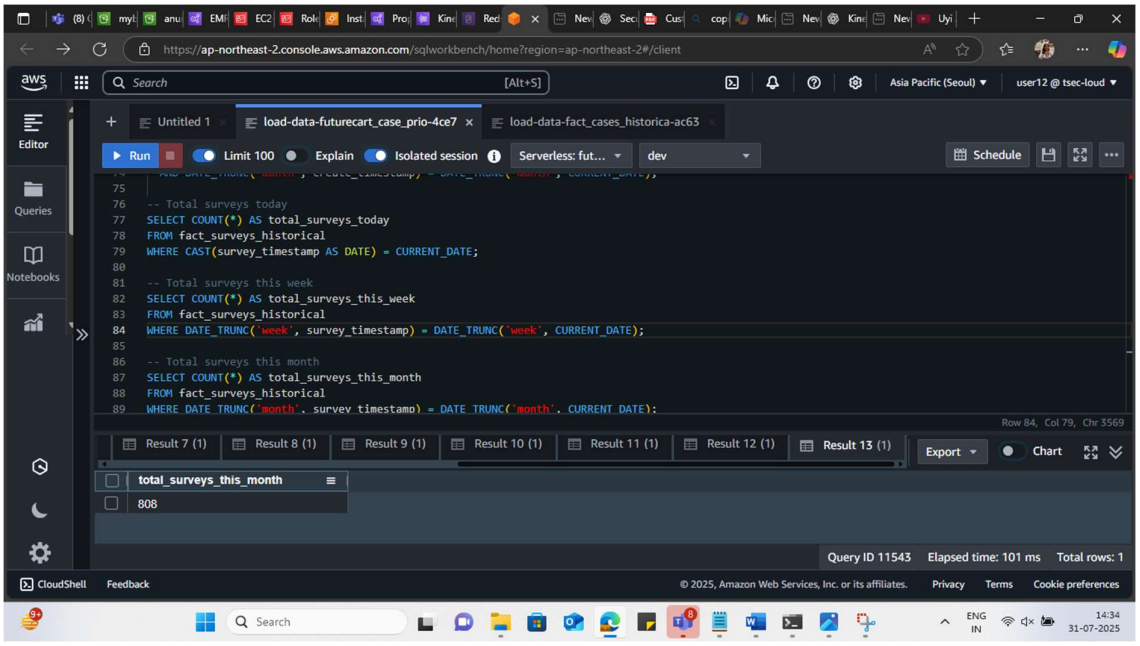

https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search [Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
Editor
Queries
Notebooks
-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 3 (1)
Result 4 (1)
Result 5 (1)
Result 6 (1)
Result 7 (1)
Result 8 (1)
Result 9 (1)
Export
Chart
closed_cases_today
8
Query ID 11523
Elapsed time: 139 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:34
31-07-2025

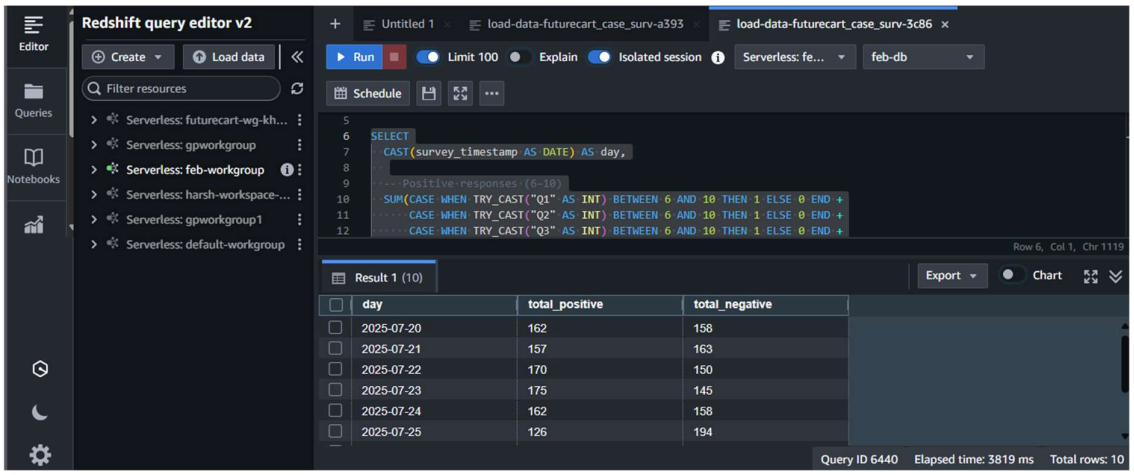

https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search
[Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Editor
Queries
Notebooks
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
75
76 -- Total surveys today
77 SELECT COUNT(*) AS total_surveys_today
78 FROM fact_surveys_historical
79 WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
80
81 -- Total surveys this week
82 SELECT COUNT(*) AS total_surveys_this_week
83 FROM fact_surveys_historical
84 WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
85
86 -- Total surveys this month
87 SELECT COUNT(*) AS total_surveys_this_month
88 FROM fact_surveys_historical
89 WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 5 (1)
Result 6 (1)
Result 7 (1)
Result 8 (1)
Result 9 (1)
Result 10 (1)
Result 11 (1)
Export
Chart
closed_cases_this_w...
58
Query ID 11527
Elapsed time: 71 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:34
31-07-2025

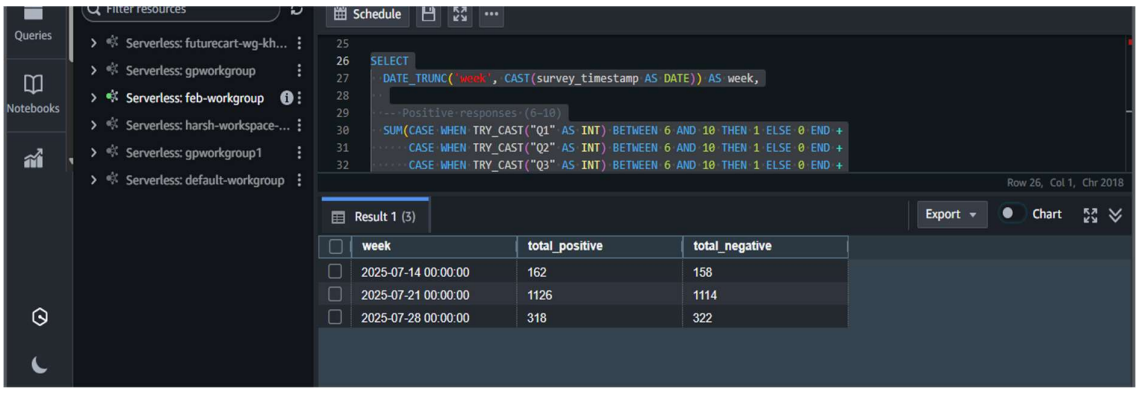

https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search
[Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Editor
Queries
Notebooks
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
75
76 -- Total surveys today
77 SELECT COUNT(*) AS total_surveys_today
78 FROM fact_surveys_historical
79 WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
80
81 -- Total surveys this week
82 SELECT COUNT(*) AS total_surveys_this_week
83 FROM fact_surveys_historical
84 WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
85
86 -- Total surveys this month
87 SELECT COUNT(*) AS total_surveys_this_month
88 FROM fact_surveys_historical
89 WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 5 (1)
Result 6 (1)
Result 7 (1)
Result 8 (1)
Result 9 (1)
Result 10 (1)
Result 11 (1)
Export
Chart
closed_cases_this_month
498
Query ID 11531
Elapsed time: 86 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:34
31-07-2025

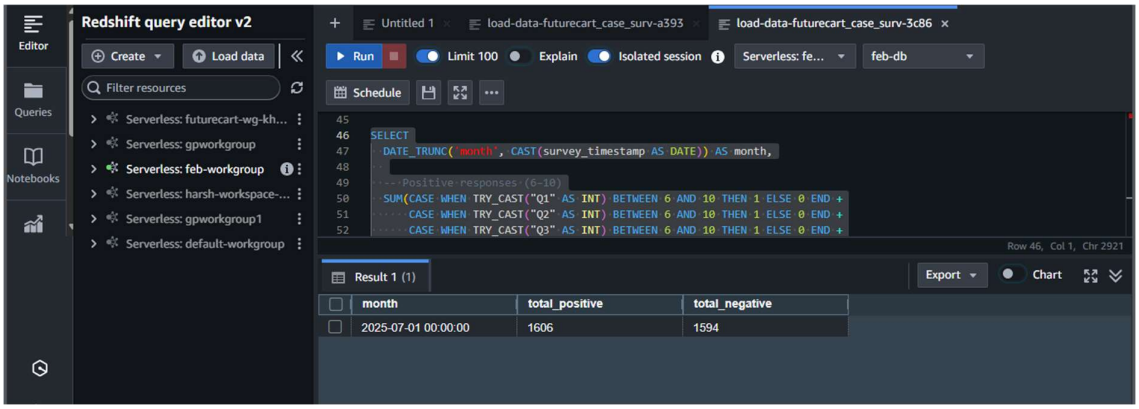

https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search
[Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
Editor
Queries
Notebooks
-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 5 (1)
Result 6 (1)
Result 7 (1)
Result 8 (1)
Result 9 (1)
Result 10 (1)
Result 11 (1)
Export
Chart
total_surveys_today
8
Query ID 11535
Elapsed time: 109 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:34
31-07-2025


https://ap-northeast-2.console.aws.amazon.com/sqlworkbench/home?region=ap-northeast-2#/client
Search
[Alt+S]
Asia Pacific (Seoul)
user12 @ tsec-loud
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
Editor
Queries
Notebooks
-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 7 (1)
Result 8 (1)
Result 9 (1)
Result 10 (1)
Result 11 (1)
Result 12 (1)
Result 13 (1)
Export
Chart
total_surveys_this_week
88


Editor
Queries
Notebooks
Untitled 1
load-data-futurecart_case_prio-4ce7
load-data-fact_cases_historica-ac63
Run
Limit 100
Explain
Isolated session
Serverless: fut...
dev
Schedule
-- Total surveys today
SELECT COUNT(*) AS total_surveys_today
FROM fact_surveys_historical
WHERE CAST(survey_timestamp AS DATE) = CURRENT_DATE;
-- Total surveys this week
SELECT COUNT(*) AS total_surveys_this_week
FROM fact_surveys_historical
WHERE DATE_TRUNC('week', survey_timestamp) = DATE_TRUNC('week', CURRENT_DATE);
-- Total surveys this month
SELECT COUNT(*) AS total_surveys_this_month
FROM fact_surveys_historical
WHERE DATE_TRUNC('month', survey_timestamp) = DATE_TRUNC('month', CURRENT_DATE);
Row 84, Col 79, Chr 3569
Result 7 (1)
Result 8 (1)
Result 9 (1)
Result 10 (1)
Result 11 (1)
Result 12 (1)
Result 13 (1)
Export
Chart
total_surveys_this_month
808
Query ID 11543
Elapsed time: 101 ms
Total rows: 1
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Redshift query editor v2
Create
Load data
Filter resources
Serverless: futurecart-wg-kh...
Serverless: gpworkgroup
Serverless: feb-workgroup
Serverless: harsh-workspace-...
Serverless: gpworkgroup1
Serverless: default-workgroup
Untitled 1
load-data-futurecart_case_surv-a393
load-data-futurecart_case_surv-3c86
Run
Limit 100
Explain
Isolated session
Serverless: fe...
feb-db
Schedule
SELECT
CAST(survey_timestamp AS DATE) AS day,
Positive responses (6-10)
SUM(CASE WHEN TRY_CAST("Q1" AS INT) BETWEEN 6 AND 10 THEN 1 ELSE 0 END +
CASE WHEN TRY_CAST("Q2" AS INT) BETWEEN 6 AND 10 THEN 1 ELSE 0 END +
CASE WHEN TRY_CAST("Q3" AS INT) BETWEEN 6 AND 10 THEN 1 ELSE 0 END +
Row 6, Col 1, Chr 1119
Result 1 (10)
Export
Chart
day
total_positive
total_negative
2025-07-20
162
158
2025-07-21
157
163
2025-07-22
170
150
2025-07-23
175
145
2025-07-24
162
158
2025-07-25
126
194
Query ID 6440
Elapsed time: 3819 ms
Total rows: 10


Queries
Notebooks
Filter resources
Serverless: futurecart-wg-kh...
Serverless: gpworkgroup
Serverless: feb-workgroup
Serverless: harsh-workspace-...
Serverless: gpworkgroup1
Serverless: default-workgroup
Schedule
SELECT
DATE_TRUNC('week', CAST(survey_timestamp AS DATE)) AS week,
Positive responses (6-10)
SUM(CASE WHEN TRY_CAST("Q1" AS INT) BETWEEN 6 AND 10 THEN 1 ELSE 0 END +
CASE WHEN TRY_CAST("Q2" AS INT) BETWEEN 6 AND 10 THEN 1 ELSE 0 END +
CASE WHEN TRY_CAST("Q3" AS INT) BETWEEN 6 AND 10 THEN 1 ELSE 0 END +
Row 26, Col 1, Chr 2018
Result 1 (3)
Export
Chart
week
total_positive
total_negative
2025-07-14 00:00:00
162
158
2025-07-21 00:00:00
1126
1114
2025-07-28 00:00:00
318
322


Redshift query editor v2
Editor
Queries
Notebooks
Create
Load data
Filter resources
Serverless: futurecart-wg-kh...
Serverless: gpworkgroup
Serverless: feb-workgroup
Serverless: harsh-workspace-...
Serverless: gpworkgroup1
Serverless: default-workgroup
Untitled 1
load-data-futurecart_case_surv-a393
load-data-futurecart_case_surv-3c86
Run
Limit 100
Explain
Isolated session
Serverless: fe...
feb-db
Schedule
45
46 SELECT
47 DATE_TRUNC('month', CAST(survey_timestamp AS DATE)) AS month,
48
49 ----Positive responses (6-10)
50 SUM(CASE WHEN TRY_CAST("Q1" AS INT) BETWEEN 6 AND 10 THEN 1 ELSE 0 END +
51 CASE WHEN TRY_CAST("Q2" AS INT) BETWEEN 6 AND 10 THEN 1 ELSE 0 END +
52 CASE WHEN TRY_CAST("Q3" AS INT) BETWEEN 6 AND 10 THEN 1 ELSE 0 END +
Row 46, Col 1, Chr 2921
Result 1 (1)
Export
Chart
month
total_positive
total_negative
2025-07-01 00:00:00
1606
1594