

Lab 8: Transactions and permissions

Course: SQL Server Database Development (ITE-5223)

Database: AdventureWorks2019

Schema Used: Reporting

Student: Yi-Chun Lien – N01745009

1. Executive Summary

This report documents the design, implementation, and validation of transactional integrity, concurrency control, and role-based access control for the AdventureWorks IoT Operations Platform.

The platform ingests high-volume telemetry from SmartBike consumer devices and industrial manufacturing sensors.

To meet enterprise requirements, the database solution ensures:

- ACID-compliant multi-step telemetry ingestion
- Protection against partial writes and corrupted packets
- Concurrency handling under high-throughput workloads
- Least-privilege access control for IoT departments
- Full auditability and error logging

The report summarizes schema creation, transactional workflow logic, failure scenario handling, concurrency behavior under different isolation levels, and validation of role-based permissions.

2. IoT Business Case Overview

AdventureWorks expanded its business into IoT-enabled devices in two major divisions:

1. SmartBike

- Consumer bicycles with GPS, cadence, temperature, crash detection, and battery sensors
- Sends telemetry every 30 seconds

2. Manufacturing IoT

- Factory-floor devices monitoring temperature, vibration, operational status
- Used for predictive maintenance and real-time monitoring

The backend database processes more than 250,000 telemetry messages per hour.

To ensure reliability and regulatory compliance (GDPR / CCPA), the system must:

- Prevent partial telemetry ingestion
- Detect invalid or corrupted sensor data
- Maintain consistent device state under concurrent operations
- Enforce strict security boundaries between IoT teams

This lab simulates a production IoT backend for both consumer and industrial devices.

3. Transaction Workflow

The stored procedure IoT.usp_ProcessTelemetryCycle performs atomic processing of a complete telemetry cycle.

It ensures that all five dependent records are created successfully, or the entire operation rolls back.

Processing Steps:

1. Validate Device Registration:

- Reject unregistered, invalid, or decommissioned devices.

2. Validate Sensor Inputs

- Temperature range -40 to 85°C
- Battery level 0–100%
- Latitude/Longitude cannot be NULL and must be within valid ranges

3. Insert Raw Telemetry into IoT.Telemetry.

4. Update LastHeartbeat in IoT.Device.

5. Insert DeviceHealth Summary

- Computes TemperatureStatus, BatteryStatus, VibrationStatus
- Marks device as Healthy/Unhealthy

6. Insert Alerts into IoT.Alert

- Low/Critical battery
- High temperature

7. Insert TelemetryAudit Record

- Processing duration
- Number of records inserted
- Status (Completed or CompletedWithAlerts)

8. Commit Transaction

Rollback Conditions

If any failure occurs:

- Missing or invalid DeviceID
- Sensor readings outside acceptable ranges
- Null or invalid coordinates
- Any conversion error or unexpected exception

Then:

- Entire transaction rolls back
- Detailed error entry is written to IoT.TelemetryErrorLog

This ensures 100% data integrity during telemetry ingestion.

4. Failure Scenarios

Three failure scenarios were executed to verify correct rollback and error logging.

Failure Case 1 – Invalid DeviceID

- DeviceID = 99999 (not registered)
- Stored procedure throws error "Device not registered"

- Result:
 1. No telemetry, health, alert, audit rows created
 2. IoT.TelemetryErrorLog contains an error entry with ErrorStep = "Validate Device"

The screenshot displays two separate sessions in SQL Server Management Studio (SSMS).

Session 1 (Top):

- Object Explorer:** Shows the connection to "localhost (SQL Server 16.0.1000.6 - Allison)\j8903".
- Query Editor:** Contains the following T-SQL code:


```

      EXEC IoT.usp_ProcessTelemetryCycle
      @DeviceID = 99999
      @Speed = 10,
      @Cadence = 80,
      @Temperature = 30,
      @BatteryLevel = 50,
      @Latitude = 43.6532,
      @Longitude = -79.3832,
      @RawPayload = N'Case1_InvalidDeviceID';
      
      SELECT TOP 5 * FROM IoT.Telemetry ORDER BY TelemetryID DESC;
      SELECT TOP 5 * FROM IoT.DeviceHealth ORDER BY HealthID DESC;
      SELECT TOP 5 * FROM IoT.Alert ORDER BY AlertID DESC;
      
```
- Messages:** Displays the error message:


```

      Msg 51001, Level 16, State 1, Procedure IoT.usp_ProcessTelemetryCycle, Line 34 [Batch Start Line 304]
      Device not registered or is decommissioned.

      Completion time: 2025-11-19T10:40:47.1953982-05:00
      
```

Session 2 (Bottom):

- Object Explorer:** Shows the connection to "localhost (SQL Server 16.0.1000.6 - Allison)\j8903".
- Query Editor:** Contains the following T-SQL code:


```

      @Speed = 10,
      @Cadence = 80,
      @Temperature = 30,
      @BatteryLevel = 50,
      @Latitude = 43.6532,
      @Longitude = -79.3832,
      @RawPayload = N'Case1_InvalidDeviceID';

      SELECT TOP 5 * FROM IoT.Telemetry ORDER BY TelemetryID DESC;
      SELECT TOP 5 * FROM IoT.DeviceHealth ORDER BY HealthID DESC;
      SELECT TOP 5 * FROM IoT.Alert ORDER BY AlertID DESC;
      SELECT TOP 5 * FROM IoT.TelemetryAudit ORDER BY AuditID DESC;
      
```
- Results:** Shows four result sets corresponding to the four tables selected in the query:
 - Telemetry: TelemetryID, DeviceID, Timestamp, Speed, Cadence, Temperature, BatteryLevel, GPSLatitude, GPSLongitude
 - Health: HealthID, DeviceID, Timestamp, IsHealthy, TemperatureStatus, BatteryStatus, VibrationStatus
 - Alert: AlertID, DeviceID, Timestamp, AlertType, Severity, Description
 - Audit: AuditID, DeviceID, Timestamp, ProcessingDuration, RecordsInserted, Status
- Messages:** Displays the success message:


```

      ✓ Query executed successfully.
      
```

Object Explorer 01_create_schema.sql - not connected lab8.sql - Ad...NVj8903 (51)*

```

313     @RawPayload = N'Case1_InvalidDeviceID';
314
315     SELECT TOP 5 * FROM IoT_Telemetry ORDER BY TelemetryID DESC;
316     SELECT TOP 5 * FROM IoT_DeviceHealth ORDER BY HealthID DESC;
317     SELECT TOP 5 * FROM IoT_Alert ORDER BY AlertID DESC;
318     SELECT TOP 5 * FROM IoT_TelemetryAudit ORDER BY AuditID DESC;
319
320     SELECT TOP 5 * FROM IoT_TelemetryErrorLog ORDER BY ErrorID DESC;
321
322
323
324
325

```

Results Messages

| ErrorID | DeviceID | Timestamp | ErrorMessage | ErrorStep | RawPayload |
|---------|----------|-------------------------|---|-----------------|-----------------------|
| 1 | 4 | 2025-11-19 15:40:47.167 | Device not registered or is decommissioned. | Validate Device | Case1_InvalidDeviceID |
| 2 | 3 | 2025-11-19 15:40:29.200 | Device not registered or is decommissioned. | Validate Device | Case1_InvalidDeviceID |

Query executed successfully.

Error List

Entire Solution 0 Errors 0 Warnings 0 Messages Build Only Search Error List

Failure Case 2 – BatteryLevel > 100

- Violates sensor rules (0–100%)
- Stored procedure throws custom error
- Result:
 1. Transaction rolled back
 2. Error logged with ErrorStep = "Validate input ranges"

Object Explorer lab8.sql - Ad...NVj8903 (51)*

```

325     VALUES ('SN-001', 'SmartBike', '1.0.0');
326
327     EXEC IoT.usp_ProcessTelemetryCycle
328         @DeviceID = 1,
329         @Speed = 20,
330         @Cadence = 60,
331         @Temperature = 25,
332         @BatteryLevel = 150,
333         @Latitude = 43.6532,
334         @Longitude = -79.3832,
335         @RawPayload = N'Case2_BatteryAbove100';
336
337
338

```

Messages

Msg 51003, Level 16, State 1, Procedure IoT.usp_ProcessTelemetryCycle, Line 43 [Batch Start Line 326]
BatteryLevel must be between 0 and 100.

Completion time: 2025-11-19T10:46:17.0657203-05:00

No issues found

Query completed with errors.

Error List

Entire Solution 0 Errors 0 Warnings 0 Messages Build Only Search Error List

```

lab8.sql - Ad...N\j8903 (51)*
332     @BatteryLevel = 150,
333     @Latitude     = 43.6532,
334     @Longitude    = -79.3832,
335     @RawPayload    = N'Case2_BatteryAbove100';
336
337     SELECT TOP 5 * FROM IoT.Telemetry ORDER BY TelemetryID DESC;
338     SELECT TOP 5 * FROM IoT.DeviceHealth ORDER BY HealthID DESC;
339     SELECT TOP 5 * FROM IoT.Alert ORDER BY AlertID DESC;
340     SELECT TOP 5 * FROM IoT.TelemetryAudit ORDER BY AuditID DESC;
341
342     SELECT TOP 5 * FROM IoT.TelemetryErrorLog ORDER BY ErrorID DESC;
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444

```

Results

| TelemetryID | DeviceID | Timestamp | Speed | Cadence | Temperature | BatteryLevel | GPSLatitude | GPSLongitude |
|-------------|----------|-----------|-------|---------|-------------|--------------|-------------|--------------|
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |

Messages

| HealthID | DeviceID | Timestamp | IsHealthy | TemperatureStatus | BatteryStatus | VibrationStatus |
|----------|----------|-----------|-----------|-------------------|---------------|-----------------|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

AlertID | DeviceID | Timestamp | AlertType | Severity | Description |

| AlertID | DeviceID | Timestamp | AlertType | Severity | Description |
|---------|----------|-----------|-----------|----------|-------------|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

AuditID | DeviceID | Timestamp | ProcessingDuration | RecordsInserted | Status |

| AuditID | DeviceID | Timestamp | ProcessingDuration | RecordsInserted | Status |
|---------|----------|-----------|--------------------|-----------------|--------|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Query executed successfully.

Error List

| Entire Solution | 0 Errors | 0 Warnings | 0 Messages | Build Only |
|-----------------|----------|------------|------------|------------|
|-----------------|----------|------------|------------|------------|

localhost (16.0 RTM) | ALLISON\j8903 (51) | AdventureWorks2019 | 00:00:00 | 0 rows


```

lab8.sql - Ad...N\j8903 (51)*
332     @BatteryLevel = 150,
333     @Latitude     = 43.6532,
334     @Longitude    = -79.3832,
335     @RawPayload    = N'Case2_BatteryAbove100';
336
337     SELECT TOP 5 * FROM IoT.Telemetry ORDER BY TelemetryID DESC;
338     SELECT TOP 5 * FROM IoT.DeviceHealth ORDER BY HealthID DESC;
339     SELECT TOP 5 * FROM IoT.Alert ORDER BY AlertID DESC;
340     SELECT TOP 5 * FROM IoT.TelemetryAudit ORDER BY AuditID DESC;
341
342     SELECT TOP 5 * FROM IoT.TelemetryErrorLog ORDER BY ErrorID DESC;
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444

```

Results

| ErrorID | DeviceID | Timestamp | ErrorMessage | ErrorStep | RawPayload | |
|---------|----------|-----------|-------------------------|---|-----------------------|-----------------------|
| 1 | 6 | 1 | 2025-11-19 15:46:17.066 | BatteryLevel must be between 0 and 100. | Validate input ranges | Case2_BatteryAbove100 |
| 2 | 5 | 1 | 2025-11-19 15:43:10.455 | Device not registered or is decommissioned. | Validate Device | Case2_BatteryAbove100 |
| 3 | 4 | 99999 | 2025-11-19 15:40:47.167 | Device not registered or is decommissioned. | Validate Device | Case1_InvalidDeviceID |
| 4 | 3 | 99999 | 2025-11-19 15:40:29.200 | Device not registered or is decommissioned. | Validate Device | Case1_InvalidDeviceID |

Messages

| Code | Description |
|------|-------------|
| | |
| | |
| | |
| | |
| | |

Query executed successfully.

Error List

| Entire Solution | 0 Errors | 0 Warnings | 0 Messages | Build Only |
|-----------------|----------|------------|------------|------------|
|-----------------|----------|------------|------------|------------|

localhost (16.0 RTM) | ALLISON\j8903 (51) | AdventureWorks2019 | 00:00:00 | 4 rows

Failure Case 3 – Missing GPS Coordinates

- Latitude = NULL
- Procedure rejects NULL GPS
- Result:
 1. The transaction was rolled back to prevent partial telemetry ingestion.
 2. TelemetryErrorLog contains correct error and RawPayload

5. Concurrency Analysis

Multiple concurrency scenarios were executed using two SQL Server sessions to observe blocking behavior and isolation level differences.

1 Blocking Scenario

A:

The screenshot shows the SSMS interface with the following details:

- Object Explorer:** Shows the connection to "localhost (SQL Server 16.0.1000.6 - Allison)\j8903".
- Query Editor:** The query window contains the following T-SQL code:


```

      359
      360
      361
      362
      363
      364
      365
      366
      367
      368
      369
      370
      SELECT TOP 5 * FROM IoT.TelemetryErrorLog ORDER BY ErrorID DESC;
      --Task 4: Concurrency Requirements for IoT Device State
      BEGIN TRAN;
      UPDATE IoT_Device
      SET Status = 'Maintenance'
      WHERE DeviceID = 1;
      
```
- Messages Window:** Displays the output:


```

      (1 row affected)
      Completion time: 2025-11-19T10:52:52.0950042-05:00
      
```
- Status Bar:** Shows "No issues found" and "Query executed successfully."
- Error List:** Shows "0 Errors", "0 Warnings", and "0 Messages".

B:

The screenshot shows the SSMS interface with the following details:

- Object Explorer:** Shows the connection to "localhost (SQL Server 16.0.1000.6 - Allison)\j8903".
- Query Editor:** The query window contains the following T-SQL code:


```

      1
      2
      3
      4
      5
      use AdventureWorks2019;
      go
      UPDATE IoT_Device
      SET Status = 'Active'
      WHERE DeviceID = 1;
      
```
- Messages Window:** Displays the output:


```

      Executing query...
      
```
- Status Bar:** Shows "Executing query..." and "0 rows".
- Error List:** Shows "0 Errors", "0 Warnings", and "0 Messages".

Observation:

B becomes BLOCKED

This simulates high-volume ingestion conflicting with maintenance actions.

2 Dirty Read – READ UNCOMMITTED

- B can see uncommitted telemetry inserts
- Demonstrates dirty reads allowed under this isolation level

The screenshot shows the SSMS interface with two tabs open: 'test.sql' and 'lab8.sql'. The 'test.sql' tab contains a script with the following content:

```

--Isolation Level
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
SELECT * FROM IoT.Telemetry WHERE DeviceID = 1;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SELECT * FROM IoT.Telemetry WHERE DeviceID = 1;

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SELECT * FROM IoT.Telemetry WHERE DeviceID = 1;

USE master;
GO
ALTER DATABASE AdventureWorks2019 SET SINGLE_USER WITH ROLLBACK IMMEDIATE;

```

The 'Results' tab displays a single row of data from the IoT.Telemetry table:

| TelemetryID | DeviceID | Timestamp | Speed | Cadence | Temperature | BatteryLevel | GPSLatitude | GPSLongitude |
|-------------|----------|-------------------------|--------|---------|-------------|--------------|-------------|--------------|
| 1 | 4 | 2025-11-19 11:18:18.465 | 100.00 | 90 | 30.00 | 80 | 43.653200 | -79.383200 |

A message bar at the bottom indicates: 'Query executed successfully.' and 'localhost (16.0 RTM) | ALLISON\j8903 (57) | AdventureWorks2019 | 00:00:00 | 1 rows'.

3 READ COMMITTED

- B waits for A to commit
- Prevents dirty reads
- Default and safest isolation level for transactional systems

The screenshot shows the SSMS interface with two tabs open: 'test.sql' and 'lab8.sql'. The 'test.sql' tab contains a script with the same content as the previous screenshot:

```

--Isolation Level
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
SELECT * FROM IoT.Telemetry WHERE DeviceID = 1;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SELECT * FROM IoT.Telemetry WHERE DeviceID = 1;

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SELECT * FROM IoT.Telemetry WHERE DeviceID = 1;

USE master;
GO
ALTER DATABASE AdventureWorks2019 SET SINGLE_USER WITH ROLLBACK IMMEDIATE;

```

The 'Results' tab is empty, showing only the column headers for the IoT.Telemetry table.

A message bar at the bottom indicates: 'Query executed successfully.' and 'localhost (16.0 RTM) | ALLISON\j8903 (57) | AdventureWorks2019 | 00:00:00 | 0 rows'.

4 REPEATABLE READ

- Prevents changes to rows read by B
- A cannot update those rows until B commits

Object Explorer

test.sql - Ad... Executing...* lab8.sql - Adv...ON\j8903 (51)*

```

18
19
--Isolation Level
20 SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
21 SELECT * FROM IoT.Telemetry WHERE DeviceID = 1;
22
23 SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
24 SELECT * FROM IoT.Telemetry WHERE DeviceID = 1;
25
26 SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
27 SELECT * FROM IoT.Telemetry WHERE DeviceID = 1;
28
29 USE master;
30 GO
31 ALTER DATABASE AdventureWorks2019 SET SINGLE_USER WITH ROLLBACK IMMEDIATE;

```

Results Messages

Executing query... | localhost (16.0 RTM) | ALLISON\j8903 (57) | AdventureWorks2019 | 00:00:00 | 0 rows

Error List

Entire Solution | 0 Errors | 0 Warnings | 0 Messages | Build Only | Search Error List

Ready

5 SNAPSHOT Isolation

- B reads a consistent version of the data
- Does not block nor get blocked
- Ideal for high-volume telemetry analytics

Object Explorer

test.sql - Ad...ON\j8903 (57)* lab8.sql - Adv...ON\j8903 (51)*

```

30
31 GO
32 ALTER DATABASE AdventureWorks2019 SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
33 GO
34 ALTER DATABASE AdventureWorks2019 SET ALLOW_SNAPSHOT_ISOLATION ON;
35 GO
36 ALTER DATABASE AdventureWorks2019 SET MULTI_USER;
37 GO
38 SET TRANSACTION ISOLATION LEVEL SNAPSHOT;
39 SELECT * FROM IoT.Telemetry WHERE DeviceID = 1;

```

Results Messages

| TelemetryID | DeviceID | Timestamp | Speed | Cadence | Temperature | BatteryLevel | GPSLatitude | GPSLongitude |
|-------------|----------|-----------|-------|---------|-------------|--------------|-------------|--------------|
|-------------|----------|-----------|-------|---------|-------------|--------------|-------------|--------------|

Query executed successfully. | localhost (16.0 RTM) | ALLISON\j8903 (57) | AdventureWorks2019 | 00:00:00 | 0 rows

Error List

Entire Solution | 0 Errors | 0 Warnings | 0 Messages | Build Only | Search Error List

Ready

6 SERIALIZABLE

- Under SERIALIZABLE isolation, the transaction in B logically restricts concurrent inserts into the same key range.
- Although the INSERT in A was not visibly blocked due to the lack of an index on DeviceID, SERIALIZABLE is still designed to prevent phantom reads and enforce the strictest ACID guarantees.
- In a properly indexed environment, the Session A insert would be blocked until Session B completes.

The screenshot shows the SSMS interface with the following details:

- Object Explorer:** Shows a connection to "localhost (SQL Server 16.0.1000.6 - Allison)\j8903".
- test.sql - Ad... Executing...***: A query window containing the following SQL code:


```

30 GO
31 ALTER DATABASE AdventureWorks2019 SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
32 GO
33 ALTER DATABASE AdventureWorks2019 SET ALLOW_SNAPSHOT_ISOLATION ON;
34 go
35 ALTER DATABASE AdventureWorks2019 SET MULTI_USER;
36 GO
37 SET TRANSACTION ISOLATION LEVEL SNAPSHOT;
38 SELECT * FROM IoT_Telemetry WHERE DeviceID = 1;
39
40 SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
41 SELECT * FROM IoT_Telemetry WHERE DeviceID = 1;
42 
```
- Error List:** Shows "Executing query..." with 0 Errors, 0 Warnings, and 0 Messages.

6. Role-Based Access Control

Roles were created to match real-world IoT departmental responsibilities while following the Least Privilege Principle.

1 IoTDeviceAgent

Allowed:

- INSERT / UPDATE on IoT.Device
- SELECT on IoT.Device

Denied:

- Any INSERT/UPDATE/DELETE on telemetry-related tables

Used for device registration staff.

2 TelemetryIngestionService

Allowed:

INSERT Telemetry, DeviceHealth, Alert, Audit

UPDATE Device.LastHeartbeat

SELECT Device

Denied:

UPDATE Device metadata (SerialNumber, FirmwareVersion, etc.)

Represents automated ingestion microservices.

3 IoTAnalyst

Allowed:

- SELECT all IoT tables

Denied:

- INSERT / UPDATE / DELETE any IoT table

Analysts can read data but cannot affect operations.

4 IoTFieldTechnician

Denied:

- SELECT DeviceOwner or any customer-linked data
- SELECT all telemetry data (Per privacy restrictions)

5 SecurityComplianceOfficer

Allowed:

- SELECT Device, DeviceOwner, Telemetry, Alerts, ErrorLog

Denied:

- All modification operations

Used for GDPR/CCPA regulatory review.

7. Permission Validation

Each role was tested using EXECUTE AS USER.

1.IoTDeviceAgent

INSERT Device → success

The screenshot shows the SSMS interface with the following details:

- Object Explorer:** Shows the connection to "localhost (SQL Server 16.0.1000.6 - Allisonj8903)" and various database objects like Databases, Security, and Server Objects.
- test.sql - Adv...Nj8903 (57) []**: The current query tab contains the following SQL code:

```
GRANT SELECT ON IoT.TelemetryErrorLog TO SecurityComplianceOfficer;
DENY INSERT, UPDATE, DELETE ON SCHEMA::IoT TO SecurityComplianceOfficer;
--Task 6 : Permission Validation
--test role: IoTDeviceAgent
EXECUTE AS USER = 'DeviceAgentUser';
-- INSERT Device
INSERT INTO IoT.Device (SerialNumber, DeviceType, FirmwareVersion)
VALUES ('TEST-DA-001', 'SmartBike', '2.0');
-- UPDATE Device
UPDATE IoT.Device
```
- Messages:** Displays the message "(1 row affected)" and the completion time "Completion time: 2025-11-19T11:40:02.0735653-05:00".
- Status Bar:** Shows "97 %", "Ln: 467 Ch: 1 SPC CRLF", and "No issues found".
- Error List:** Shows "Entire Solution" with 0 Errors, 0 Warnings, and 0 Messages.

INSERT Telemetry → denied

The screenshot shows the SSMS interface with the following details:

- Object Explorer:** Shows the connection to "localhost (SQL Server 16.0.1000.6 - Allison)\j8903" and various database objects like Databases, Security, Server Objects, Replication, Always On High Availability, Management, Integration Services Catalogs, and SQL Server Agent.
- Query Editor:** The current tab is "test.sql - Adv...SON\j8903 (57)". The code is as follows:

```
434 EXECUTE AS USER = 'DeviceAgentUser';
435
436 -- INSERT Device
437 INSERT INTO IoT_Device (SerialNumber, DeviceType, FirmwareVersion)
438 VALUES ('TEST-DA-001', 'SmartBike', '2.0');
439
440 -- UPDATE Device
441 UPDATE IoT_Device
442 SET FirmwareVersion = '2.1'
443 WHERE SerialNumber = 'TEST-DA-001';
444
445 --INSERT Telemetry
446 INSERT INTO IoT_Telemetry (DeviceID, [Timestamp], Speed)
447 VALUES (1, SYSDATETIME(), 10);
448
449 --UPDATE Telemetry
450 UPDATE IoT_Telemetry
451 SET Speed = 99
```

- Messages:** A red error message is displayed: "Msg 229, Level 14, State 8, Line 446 The INSERT permission was denied on the object 'Telemetry', database 'AdventureWorksLT2019', schema 'IoT'."
- Completion time:** 2025-11-15T12:06:07.5783896-05:00
- Status Bar:** Shows "97 %", "Ln: 446 Ch: 1 SPC CRLF".
- Bottom Status Bar:** Shows "No issues found", "Query completed with errors.", "localhost (16.0 RTM) | ALLISON\j8903 (51) | AdventureWorksLT2019 | 00:00:00 | 0 rows", "Ln: 5 Ch: 1 TABS MIXED".
- Error List:** Shows "Entire Solution" with 0 Errors, 0 Warnings, and 0 Messages. It includes columns for Code, Description, Project, File, and Line.

2, TelemetryIngestionService
INSERT Telemetry → success

object Explorer ▾ × test.sql - Adv...SONY 8903

```
SET XACT_ABORT ON
--TEST DA-001
--SerialNumber = 'TEST-DA-001'
--INSERT Telemetry
INSERT INTO IoT_Telemetry (DeviceID, [Timestamp], Speed)
VALUES (1, SYSDATETIME(), 10);

--UPDATE Telemetry
UPDATE IoT_Telemetry
SET Speed = 99
WHERE TelemetryID = 1;

REVERT;
--Role 2 : TelemetryIngestionService
EXECUTE AS USER = 'IngestionUser';

--INSERT INTO IoT_Telemetry (DeviceID, [Timestamp], Speed)
VALUES (1, SYSDATETIME(), 15);
```

Messages

(1 row affected)

Completion time: 2025-11-19T12:21:11.9521229-05:00

No issues found

Query executed successfully.

localhost (16.0 RTM) | ALLISON\j8903 (51) | AdventureWorks2019 | 00:00:00 | 0 rows

UPDATE FirmwareVersion → denied

Object Explorer

localhost (SQL Server 16.0.1000.6 - Allisonj893)

```

497 UPDATE IoT_Telemetry
498 SET Speed = 99
499 WHERE TelemetryID = 1;
500
501 REVERT;
502 --Role 2 : TelemetryIngestionService
503 EXECUTE AS USER = 'IngestionUser';
504
505 INSERT INTO IoT_Telemetry (DeviceID, [Timestamp], Speed)
506 VALUES (1, SYSUTCDATETIME(), 15);
507
508 UPDATE IoT_Device
509 SET LastHeartbeat = SYSUTCDATETIME()
510 WHERE DeviceID = 1;
511
512 UPDATE IoT_Device
513 SET FirmwareVersion = '999.9'
514 WHERE DeviceID = 1;

```

97% 99+ ▲ 0 ↑ ↓ Ln: 510 Ch: 1 SPC CRLF

Messages

Msg 230, Level 14, State 1, Line 511
The UPDATE permission was denied on the column 'FirmwareVersion' of the object 'Device', database 'AdventureWorks2019', schema 'IoT'.

Completion time: 2025-11-19T12:23:32.0035835-05:00

97% No issues found Ln: 5 Ch: 1 TABS MIXED

Error List

Query completed with errors. localhost (16.0 RTM) | ALLISONj8903 (51) | AdventureWorks2019 | 00:00:00 | 0 rows

Entire Solution 0 Errors 0 Warnings 0 Messages Build Only Search Error List

Code Description Project File Line

3.IoTAnalyst

SELECT → SUCCESS

Object Explorer

localhost (SQL Server 16.0.1000.6 - Allisonj893)

```

500
501 --Role 2 : TelemetryIngestionService
502 EXECUTE AS USER = 'IngestionUser';
503
504 INSERT INTO IoT_Telemetry (DeviceID, [Timestamp], Speed)
505 VALUES (1, SYSUTCDATETIME(), 15);
506
507 UPDATE IoT_Device
508 SET LastHeartbeat = SYSUTCDATETIME()
509 WHERE DeviceID = 1;
510
511 UPDATE IoT_Device
512 SET FirmwareVersion = '999.9'
513 WHERE DeviceID = 1;
514
515 REVERT;
516 --Role 3 : IoTAnalyst (Read-Only)
517 EXECUTE AS USER = 'AnalystUser';
518 SELECT TOP 5 * FROM IoT_Telemetry;

```

97% 99+ ▲ 0 ↑ ↓ Ln: 517 Ch: 1 SPC CRLF

Results

| TelemetryID | DeviceID | Timestamp | Speed | Cadence | Temperature | BatteryLevel | GPSLatitude | GPSLongitude |
|-------------|----------|-------------------------|--------|---------|-------------|--------------|-------------|--------------|
| 1 | 4 | 2025-11-19 11:18:18.465 | 100.00 | 90 | 30.00 | 80 | 43.653200 | -79.383200 |
| 2 | 5 | 2025-11-19 17:21:11.940 | 15.00 | NULL | NULL | NULL | NULL | NULL |

Query executed successfully. localhost (16.0 RTM) | ALLISONj8903 (51) | AdventureWorks2019 | 00:00:00 | 2 rows

Error List

Entire Solution 0 Errors 0 Warnings 0 Messages Build Only Search Error List

Code Description Project File Line

INSERT → denied

The screenshot shows the SQL Server Management Studio interface. In the Object Explorer, a connection to 'localhost (SQL Server 16.0.1000.6 - Allison\j8903)' is selected. The 'Databases' node is expanded, showing 'AdventureworksLT', 'AdventureworksDW', and 'Adventureworks'. In the center pane, a query window titled 'test.sql - Adv-SON(j8903 (57))' is open, displaying a series of INSERT and UPDATE statements for the IoT database. The final statement is an INSERT into 'IoT.Telemetry'. In the bottom-left pane, the 'Messages' tab is active, showing a red error message: 'Msg 229, Level 14, State 5, Line 519 The INSERT permission was denied on the object 'Telemetry', database 'AdventureWorks2019', schema 'IoT'. Completion time: 2025-11-19T12:26:13.2210293-05:00'. The status bar at the bottom right indicates 'Ln: 5 Ch: 1 SPC CRLF'.

4. IoTFieldTechnician

SELECT Device → denied

Object Explorer

Connect

localhost (SQL Server 16.0.1000.6 - Allison)\j8909

- Databases
- Security
- Server Objects
- Replication
- Always On High Availability
- Management
- Integration Services Catalogs
- SQL Server Agent (Agent XPs disabled)
- XEvent Profiler

```
314      ALERT;
315  --Role 3: IoTAnalyst (Read-Only)
316  EXECUTE AS USER = 'AnalystUser';
317  SELECT TOP 5 * FROM IoT.Telemetry;
318
319  ✓ INSERT INTO IoT.Telemetry (DeviceID, [Timestamp], Speed)
320    VALUES (1, SYSDATETIME(), 22);
321
322  ✓ UPDATE IoT.DeviceHealth
323    SET BatteryStatus = 'Low'
324    WHERE DeviceID = 1;
325    revert;
326  --Role 4: IoTFieldTechnician
327  EXECUTE AS USER = 'TechUser';
328  SELECT TOP 5 * FROM IoT.Device;
```

97% 99+ ⚠ 0 ↑ ↓ ↻ 🔍

Messages

Mag 229, Level 14, State 5, Line 528
The SELECT permission was denied on the object 'Device', database 'AdventureWorks2019', schema 'IoT'.

Completion time: 2025-11-19T12:28:21.6833502-05:00

97% ✅ No issues found

⚠ Query completed with errors.

Error List

Entire Solution 0 Errors 0 Warnings 0 Messages Build Only

Project File Line

Ready

5. SecurityComplianceOfficer

SELECT Audit → success

Object Explorer

localhost (SQL Server 16.0.1000.6 - Allison\j8903)

test.sql - Adv...SONY8903 (57) lab8.sql - Ad...Nv8903 (51)*

```

523 SET DATASOURCES OWN
524 WHERE DeviceID = 1;
525 revert;
526 --Role 4:IoTFieldTechnician
527 EXECUTE AS USER = 'TechUser';
528 SELECT TOP 5 * FROM IoT_Device;
529
530 SELECT TOP 5 * FROM IoT_Telemetry;
531 REVERT;
532 --Role 5:SecurityComplianceOfficer
533 EXECUTE AS USER = 'ComplianceUser';
534 SELECT TOP 5 * FROM IoT_TelemetryAudit;
535
536
537

```

Results

| AuditID | DeviceID | Timestamp | ProcessingDuration | RecordsInserted | Status |
|---------|----------|-----------|--------------------|-----------------|--------|
| | | | | | |

Query executed successfully.

Error List

localhost (16.0 RTM) | ALLISON\j8903 (51) | AdventureWorks2019 | 00:00:00 | 0 rows

INSERT ErrorLog → denied

Object Explorer

localhost (SQL Server 16.0.1000.6 - Allison\j8903)

test.sql - Adv...SONY8903 (57) lab8.sql - Ad...Nv8903 (51)*

```

523 SET DATASOURCES OWN
524 WHERE DeviceID = 1;
525 revert;
526 --Role 4:IoTFieldTechnician
527 EXECUTE AS USER = 'TechUser';
528 SELECT TOP 5 * FROM IoT_Device;
529
530 SELECT TOP 5 * FROM IoT_Telemetry;
531 REVERT;
532 --Role 5:SecurityComplianceOfficer
533 EXECUTE AS USER = 'ComplianceUser';
534 SELECT TOP 5 * FROM IoT_TelemetryAudit;
535
536 INSERT INTO IoT_TelemetryErrorLog (ErrorMessage)
537 VALUES ('fake error');
538
539
540

```

Messages

Msg 229, Level 14, State 5, Line 536
The INSERT permission was denied on the object 'TelemetryErrorLog', database 'AdventureWorks2019', schema 'IoT'.

Completion time: 2025-11-19T12:31:07.7155298-05:00

Query completed with errors.

Error List

localhost (16.0 RTM) | ALLISON\j8903 (51) | AdventureWorks2019 | 00:00:00 | 0 rows

All tests confirmed permissions were correctly enforced.

8. Conclusion

This lab successfully implemented a production-grade IoT backend aligned with enterprise database engineering standards.

Key achievements include:

1.ACID-Compliant Telemetry Pipeline

Prevents partial sensor data ingestion and ensures reliable device health computation.

2.Advanced Concurrency Control

Blocking, dirty reads, repeatable reads, and snapshot isolation behaviors were validated across real SQL Server sessions.

3. Secure Role-Based Access Control

Strict least-privilege permissions ensure compliance with privacy regulations (GDPR/CCPA).

4. Professional Logging & Auditing

TelemetryErrorLog and TelemetryAudit provide operational traceability for monitoring,

debugging, and compliance review.

Overall, the system is secure, reliable, and scalable for large-scale real-time IoT deployments.