Term Project Iteration 4

Phillip Escandon
escandon@bu.edu

Project Direction / Overview

Aerospace Camera System Error Logs, Maintenance Logs, Configuration Files-

During the process of building and integrating a camera system and all of its subparts, many log files are captured but not widely used. When called upon to review the logs, only a few team members have the background and historical knowledge to review and decipher any issues as well as suggest a course of action to rectify these issues.

My proposed database will contain log files and the associated describing data for each of these camera systems. I have collected approximately 9000 of these logs dating back to 2012. My goal would be a database that could quickly create a report(s) to:

1. Give a historical background and timeline of the camera, including SW versions, Configuration of the camera system when it was delivered to a customer and common failures observed during testing.
2. Give a historical background and normalized boot sequence for a given camera system.
3. Give a historical background of *failures* and some basic configuration and state that the camera was in during a failure.

Some of these items may seem trivial,but the collection of these logs and *quickly* deciphering them has always been an issue.

This database could potentially be used by four (4) teams.

The data to be captured and used in the database are the logs that are particular to each product.

Three particular logs will be used:

Errorlog.log
Maintenance.log
Sensor configuration

1. Log Files

Error Log

In an abstract framework, the logs can be considered to consist of three sections:

- **Boot Section** - describes in terse, software terms, the boot sequence of the product.
- **Start Up Built In Test (SBIT)**- Once boot is completed, the camera synchronizes with a GPS clock and a built in test is conducted. It describes the status of the various subsystems in the product.
- **Plan-** describes the tasks allocated to the product and records the user interaction with the product until the product is shut down.

Hundreds of these logs are collected for each system. Each system has a distinct **SENSOR ID**. The logs are always called 'errorlog.log' or 'maint.log'. Each errorlog will contain the Boot and SBIT section, but not necessarily the mission section. The maintenance log will contain Start times, end time, PlanID and SBIT Status.

Once the product is fielded for a customer and used, the Plan section will contain valid recorded data.

Current usage of these logs is minimal. Thousands of these logs exist, but they are rarely examined or mined for details and analysis. A shortened version of this log exists as a 'maintenance log' and is primarily used by the Field Service Team to quickly debug issues.

Maintenance Log

This is a simple version of the error log that only contains Plan ID, SBIT failures, startup and shutdown times. Everything in this log can be found in the Error log.

Configuration File

Each sensor has it's very own configuration file. This contains values that were tuned during the integration phase as well as the final focus values for the various field of views.

Interest

I have seen and used these logs but feel that they are being overlooked. Sloppy warehousing, incomplete datasets and timeframes, no real direction and no plan to use these logs or versions of these logs in upcoming projects.

For one particular vexing issue, I was given a small dataset of roughly 50 files- a list of keywords and general instruction as to see if I could figure out what was happening.

After some initial parsing and cleaning, I was still left with smaller and still very obtuse files. I could see what was happening, and could verbally explain but the team required context. They required a story to go along with this data. A parsing of the files did not provide the needed story and background.

The general questions that should be answered are:

- What Sensor had an issue?
  Sensor ID
- When did it have an issue?
  Date
- What is the SW load for each component?
  SW Versioning information
- Did it pass SBIT?
  Test Report - state of the system before mission
- What was tasked in a mission?
  What was planned?
- What failed?
  What planned task did not work?
- What passed?
  What planned task did work?
- What keys were pressed on the system?
  What buttons did the user press during operation?

1.1 Use Cases

Integration Team

The database could be used to identify issues with the product prior to release and shipment of the product. This team also captures data related to tests that are conducted in the 3 month build cycle, as well as a final configuration file.

Use Case: An Integration Engineer completes her test and is ready to update the database with her configuration file. The file will be added to the database and a report will be generated that compares each field with the corresponding fields from previous sensor ids. If each value is within +- 3 standard deviations from the mean of all previous values, then it is accepted. If a value exceeds 3 STD from the mean, the value is flagged as an outlier for further analysis. This will occur for each of the 6 tests completed: Controls, Line of Sight, Thermal, Auxiliary, Imaging, Functional Baseline.

Customer Engineering / Chief Engineers Group

By keying and reflecting on a complete report, or series of reports generated by the database, a coherent story that describes successes and failures for individual systems can be generated. An engineer assigned to support country X, could quickly find all sensors delivered to X, SW versioning information and historical data. Decisions as to the exact SW load delivered and any required updates could be tracked and seen using this database. New log files could be inserted and reports generated to provide actionable direction to correct issues.

Use Case: A Chief Engineer (CE) for Greece must gather a detailed report of all systems delivered to that country. The cameras were delivered 8 years ago.

The CE would search the database for ‘Greece’ and find 8 sensors delivered throughout a two year period. A report would indicate the IDs of the sensors delivered as well as software builds.

Using the only customer logs, a rudimentary Elapsed Time of usage could be calculated for each sensor.

Product Support / Quality Control

Failures of the product could be captured in the database in the form of a test report generated from the SBIT section of the errorlog.log. The database could also be used as a final sell-off deliverable as proof that the product has indeed passed all required tests and document corrective actions that rectified failed tests. The serial numbers and of major components could be captured and used by the Production and Quality team.

Use Case: The Quality team has just informed the customer that the Functional Baseline test has passed and the system will be ready for final acceptance testing.

The quality engineer would search for ‘Greece 2’ and find the ‘status’ report indicating the sensor ID, tests conducted, Built in Test status and software versions of the AIB, RMS, SCU and GEDI computer boards. This report would go into the sell-off documentation stating that the system has been tested and has passed all requirements.

1.2 Fields

Field	**What is Stores**	**Why it's needed**
Error Log.Sensor ID Boot.SensorID SBIT.SensorID Plan.SensorID Camera.SensorID	Stores the ID of the individual camera	Each camera system is unique and has this unique identifier.
Boot.TimeDate SBIT.TimeDate Plan.TimeDate IntegrationTest.TimeDate	Time and Date of the camera operation	This is the time and date field from the logs in Unix Epoch time. This will be converted to GMT time for the database.
Plan.PlanID	Identifies the specific planned use and configuration of the camera	This identifies the specific plan that was used during the operation. This is not a unique field.
Boot.RMSVersion Boot.SCUVersion Boot.GEDIVersion Boot.AIBVersion	Software version of the four computer boards	This is the SW build version for each computer board. Usually does not change but is very helpful for historical reports.
Boot.Message SBIT.Message Plan.Message	Message string from the logs	String message from the error log. This usually gives the details of pass / fail actions.
SBIT.Status	Boolean for testing purposes	This will give us a quick overview of the status of a test / retest sequence.
Boot.TimeSync	The time that the GPS and computer times were synchronized	This gives an indication of system boots. If more than one Time Synchronization is seen, this could indicate a problem where the camera unintentionally restarted.
ErrorLog.filename Boot.filename SBIT.filename Plan.filename	Unique filename created from the errorlog	The errorlog must be renamed as a unique entity. This unique filename will be used to identify the parent of the three (Boot, SBIT, Plan) instances generated.
Boot.message	Variable length message from the	Distinct message from the errorlog

SBIT.message Plan.message	errorlog section	section that is used to give greater context.

Field	**What is stores**	**Why it's needed**
RollResolver.Devicepub.offset	Offset for the Roll Resolver on camera	Historical Data - needed for comparison to other systems
RollResolverConfig.offset	Resolver configuration offset	Historical Data - needed for comparison to other systems
PitchResolver.Devicepub.offset	Offset for the Pitch Resolver on camera	Historical Data - needed for comparison to other systems
PitchResolverConfig.offset	Resolver configuration offset	Historical Data - needed for comparison to other systems
BS.nf1bm0 ConfigurationFile.Filter1	Backscan Z transform filter value	These values are constantly looked up and reused. Historical Data - needed for comparison to other systems
ConfigurationFile.Filter2 BS.nf1bm1	Backscan Z transform filter value	These values are constantly looked up and reused. Historical Data - needed for comparison to other systems
ConfigurationFile.Filter3 BS.nf1bm2	Backscan Z transform filter value	These values are constantly looked up and reused. Historical Data - needed for comparison to other systems
ConfigurationFile.Filter4 BS.df1bm0	Backscan Z transform filter value	These values are constantly looked up and reused. Historical Data - needed for comparison to other systems
ConfigurationFile.Filter5 BS.df1bm1	Backscan Z transform filter value	These values are constantly looked up and reused.

		Historical Data - needed for comparison to other systems.
ConfigurationFile.Filter6 BS.df1bm2	Backscan Z transform filter value	Historical Data - needed for comparison to other systems

1.3 Summary and Reflection

2.1 Structural Database Rules for Entities

A Sensor Family is made up of Cameras
Camera belongs to a Sensor Family

Sensor Family	
SensorID	Decimal 3
is_DB110	BOOL
is_MS110	BOOL
Is_TACSAR	BOOL

Camera	
SensorID	Decimal 3
Customer	VARCHAR(255)

A Camera may contain one configuration file.
A Configuration file belongs to one camera

Camera	
SensorID	Decimal 3
Customer	VARCHAR(255)

Configuration File	
SensorID	Decimal 3
RollResDeviceOffset	Decimal 12
RollResConfiOffset	Decimal 12
PitchResDeviceOffset	Decimal 12
PitchResDeviceOffset	Decimal 12
BS_nf1bm0	Decimal 12
BS_nf1bm1	Decimal 12
BS_nf1.bm2	Decimal 12
BS_df1bm0	Decimal 12
BS_df1bm1	Decimal 12
BS_df1bm2	Decimal 12

An Error Log may generates one or more Boot Logs

A Boot Log is generated by one Error Log

Error Log	
SensorID	Decimal 3
FileName	VARCHAR(50)
RMSVersion	VARCHAR(50)
SCUVersion	VARCHAR(50)
GEDIVersion	VARCHAR(50)
AIBVersion	VARCHAR(50)

Boot Log	
SensorID	Decimal 3
Time	DATE
FileName	VARCHAR(50)
Message	VARCHAR(255)

An Error Log may generates one or more SBIT Logs
An SBIT Log is generated by one Error Log

Error Log	
SensorID	Decimal 3
FileName	VARCHAR(50)
RMSVersion	VARCHAR(50)
SCUVersion	VARCHAR(50)
GEDIVersion	VARCHAR(50)
AIBVersion	VARCHAR(50)

SBIT Log	
SensorID	Decimal 3
Time	DATE
FileName	VARCHAR(50)
Test	VARCHAR(12)
Status	VARCHAR(12)
Message	VARCHAR(255)

An Error log may generates one Plan Log
A Plan Log is generated by one Error Log

Error Log	
SensorID	Decimal 3
FileName	VARCHAR(50)
RMSVersion	VARCHAR(50)
SCUVersion	VARCHAR(50)
GEDIVersion	VARCHAR(50)
AIBVersion	VARCHAR(50)

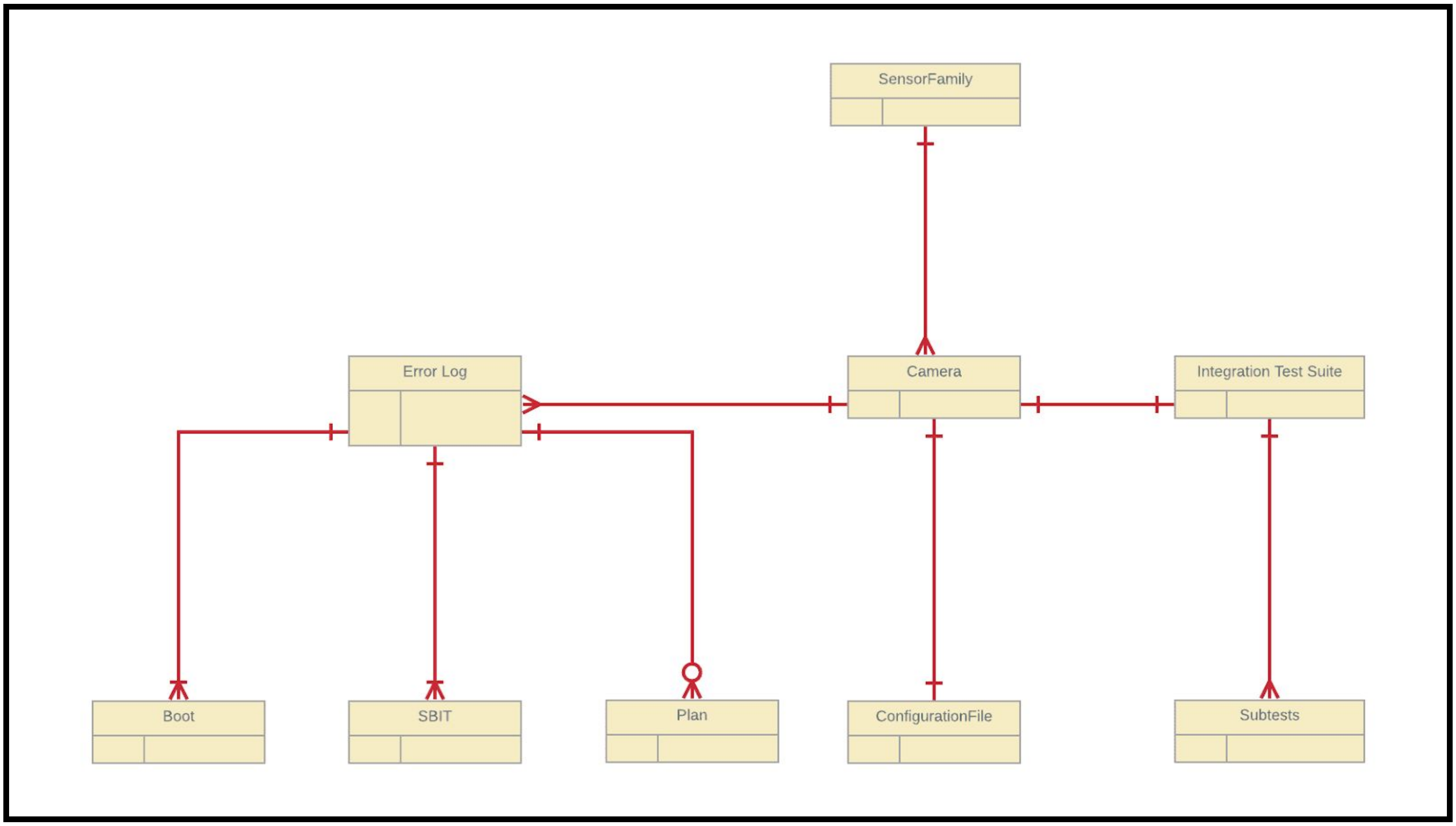
Plan Log	
SensorID	Decimal 3
Time	DATE
FileName	VARCHAR(50)
Tasks	VARCHAR(50)
Datalink	BOOL
PlanID	VARCHAR(50)
Message	VARCHAR(255)

Integration Tests is composed of many subtests.
One subtest belongs to one Integration Test.

Integration Tests	
SensorID	Decimal 3
TestID	DECIMAL 3
Subtest	VARCHAR(50)
Completion_Percent	DECIMAL 3
DataLocation	VARCHAR(255)

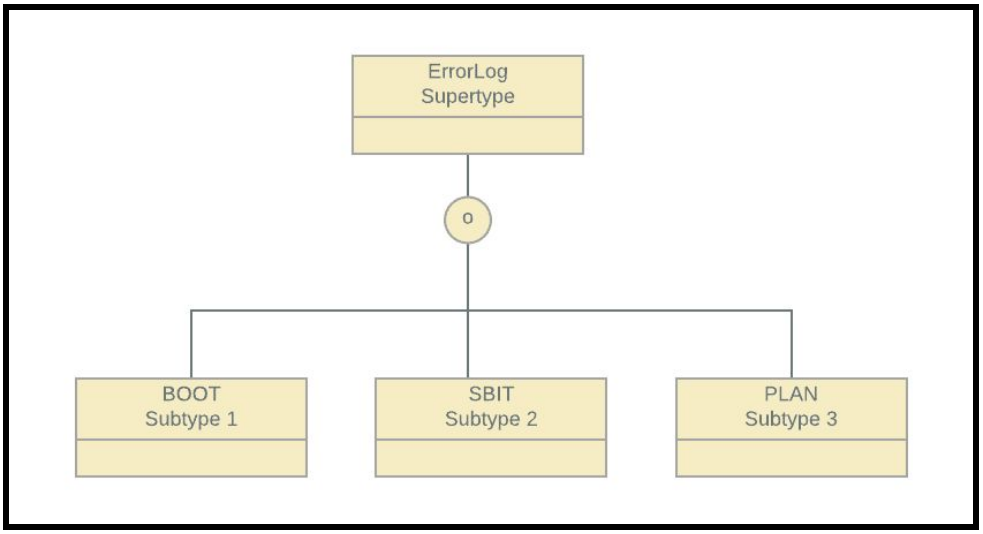
SubTests	
SensorID	Decimal 3
TestID	Decimal 3
SubtestID	Decimal 3
Status	BOOL

2.2 Conceptual ERD Diagram



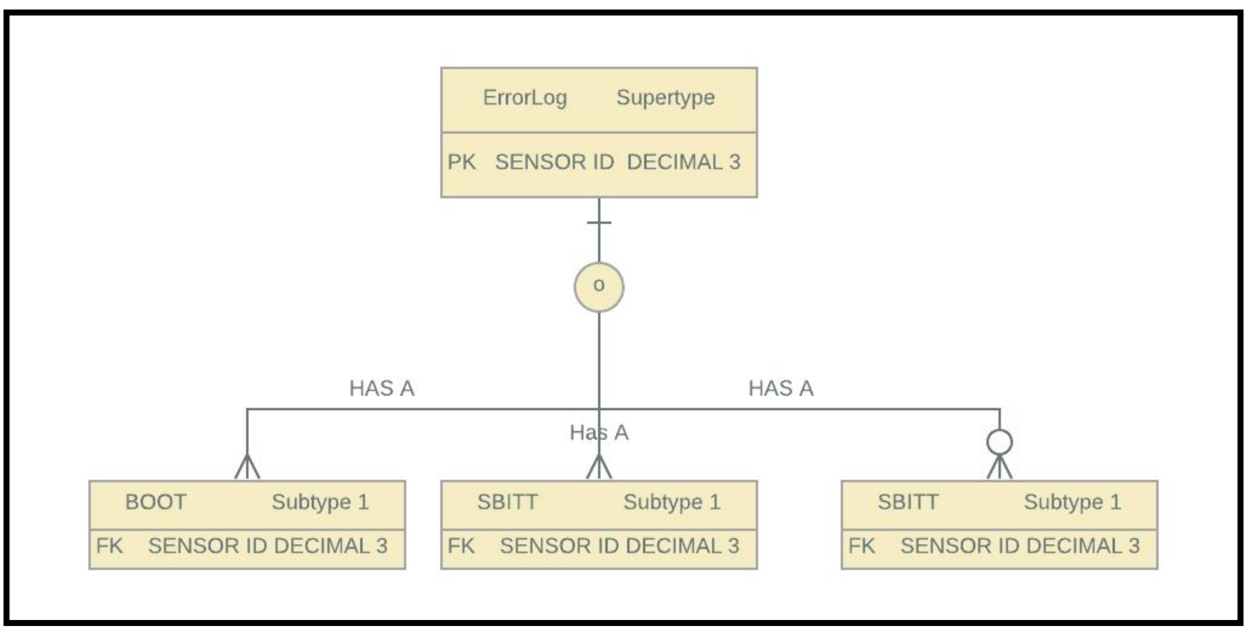
3.1 Specialization - Generalization Structural Database Rules

An Error log has a Boot log one of these or several of these.
An Error log has a SBIT log one of these or several of these.
An Error log has a Plan log,one of these or several of these.



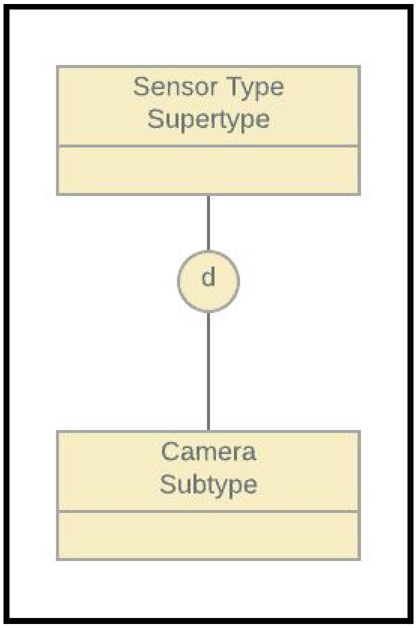
Conceptual

3.2 Initial DBMS Physical ERD

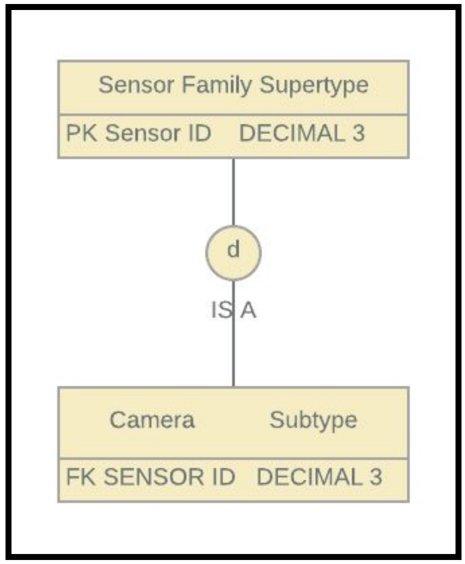


DBMS Physical

A Sensor Type is a Camera type and only one of these.



Conceptual



Physical

3.2 Specialization - Generalization Use Case:

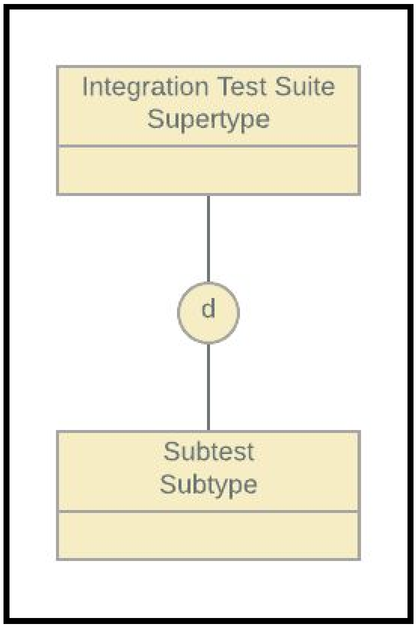
Chief Engineers (CE) Group Use case (old)

1. CE must gather information for a camera failure that occurred on a camera delivered to a customer.
2. CE would search database for customer name.
3. A report would indicate the camera IDs and software builds delivered for each camera and elapsed time of use for each camera.

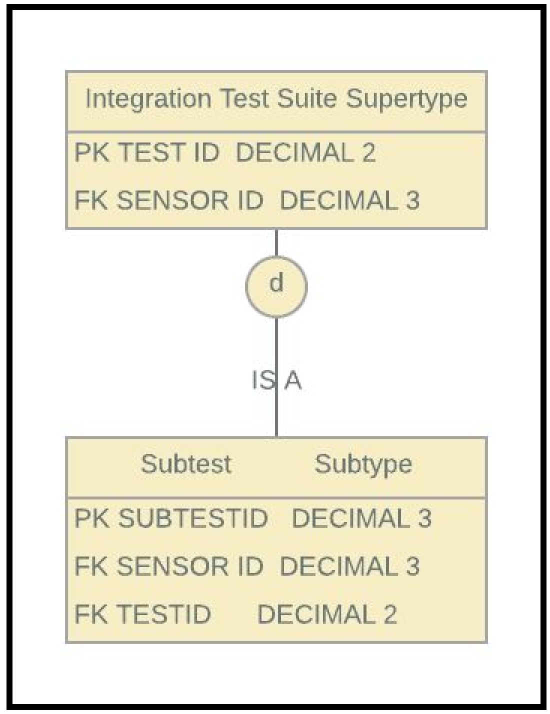
CE Group use case (new)

1. CE must gather information for a camera failure that occurred in the field
2. CE would search the database for a specific camera type for a customer name.
3. A report would indicate the camera ID and software builds delivered for each camera and elapsed time of use for each camera.

Integration Test is a Subtest and only one test.



Conceptual DBMS



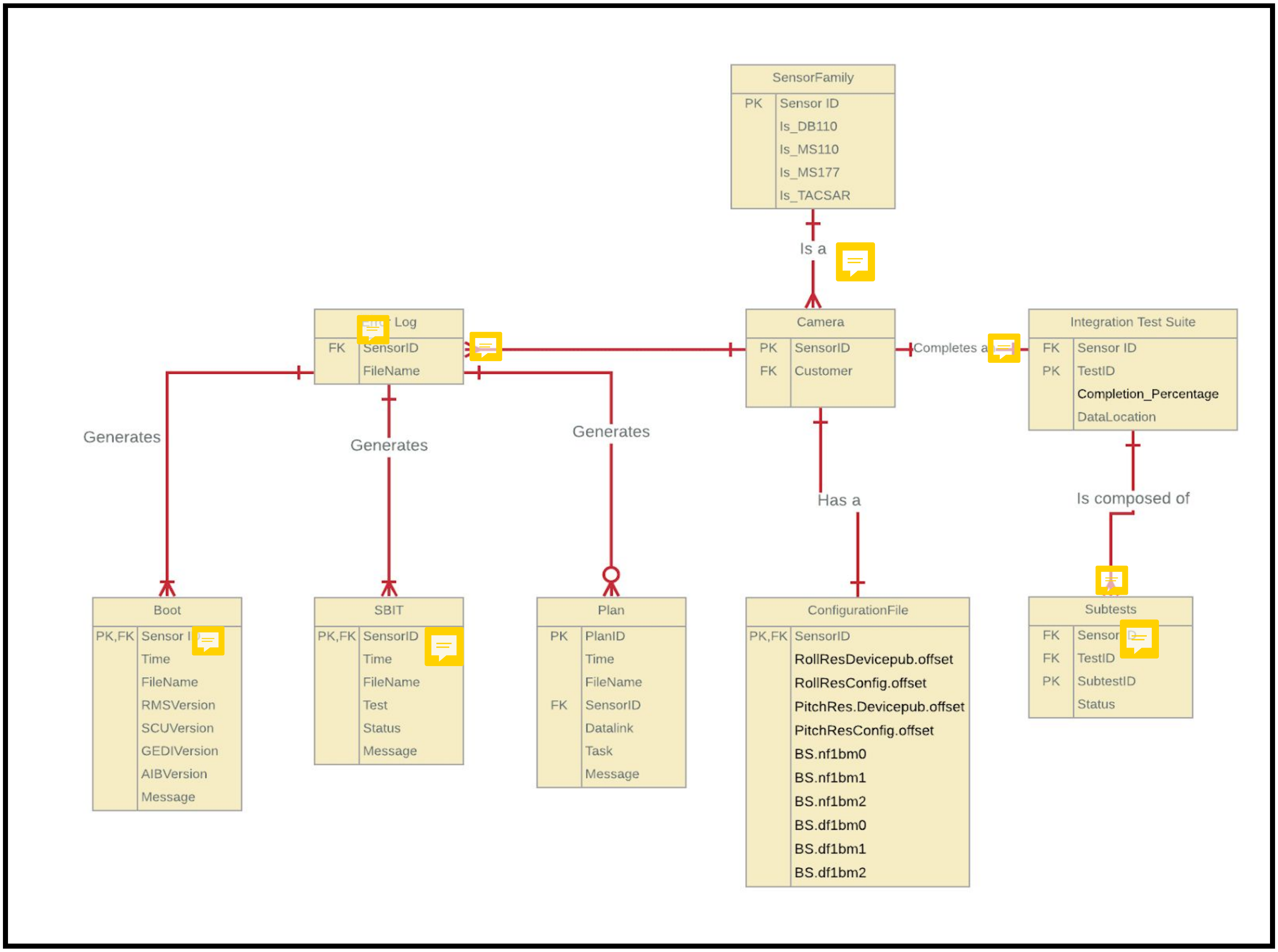
Physical DBMS

4. Full DBMS Physical ERD

Attributes

Table	Attribute	Data Type	Reasoning
Error Log	SensorID	Decimal(3)	SensorID will be a number between 1 - 500
Error Log	FileName	Varchar(50)	Filenames will be 'date_errorlog_sensorid.log'
Boot	SensorID	Decimal(3)	SensorID will be a number between 1 - 500
Boot	Time	Decimal(16)	Epoch Time
Boot	FileName	Varchar(50)	Filenames will be 'date_errorlog_sensorid.log'
Boot	RMSVersion	Varchar(255)	Usually a long string such as DB110_IQAF_MAIN_REL16.3_20140925
Boot	SCUVersion	Varchar(255)	Usually a long string such as DB110_IQAF_MAIN_REL16.3_20140926
Boot	GEDIVersion	Varchar(255)	Usually a long string such as DB110_IQAF_MAIN_REL16.3_20140927
Boot	AIBVersion	Varchar(255)	Usually a long string such as DB110_IQAF_MAIN_REL16.3_20140928
Boot	Message	Varchar(255)	Usually a long string such as DB110_IQAF_MAIN_REL16.3_20140929
SBIT	SensorID	Decimal(3)	SensorID will be a number between 1 - 500
SBIT	Time	Decimal(16)	Epoch Time
SBIT	FileName	Varchar(50)	Filenames will be 'date_errorlog_sensorid.log'
SBIT	Test	Varchar(25)	Epoch Time
SBIT	Status	bit	0 = Fail, 1 = Pass
SBIT	Message	Varchar(255)	String with additional SBIT information
Plan	PlanID	Varchar(50)	Usually something like 'Date_pilotName' or 'Date_Area'
Plan	SensorID	Decimal(3)	SensorID will be a number between 1 - 500
Plan	Time	Decimal(16)	Epoch Time
Plan	FileName	Varchar(50)	Filenames will be 'date_errorlog_sensorid.log'
Plan	Tasks	Decimal(3)	Maximum number < 100
Plan	Datalink	bit	True or False to indicate is DL was in use
ConfigurationFile	SensorID	Decimal(3)	SensorID will be a number between 1 - 500
ConfigurationFile	RollResDevicepub.offset	Decimal(2,8)	Values tend to look like x.xxxxxxxx
ConfigurationFile	RollResConfig.offset	Decimal(2,8)	Values tend to look like x.xxxxxxxx
ConfigurationFile	PitchRes.Devicepub.offs	Decimal(2,8)	Values tend to look like x.xxxxxxxx

	et		
ConfigurationFile	PitchResConfig.offset	Decimal(2,8)	Values tend to look like x.xxxxxxxx
ConfigurationFile	BS.nf1bm0	Decimal(2,8)	Values tend to look like x.xxxxxxxx
ConfigurationFile	BS.nf1bm1	Decimal(2,8)	Values tend to look like x.xxxxxxxx
ConfigurationFile	BS.nf1bm2	Decimal(2,8)	Values tend to look like x.xxxxxxxx
ConfigurationFile	BS.df1bm0	Decimal(2,8)	Values tend to look like x.xxxxxxxx
ConfigurationFile	BS.df1bm1	Decimal(2,8)	Values tend to look like x.xxxxxxxx
ConfigurationFile	BS.df1bm2	Decimal(2,8)	Values tend to look like x.xxxxxxxx
Camera	SensorID	Decimal(3)	SensorID will be a number between 1 - 500
Camera	Customer	Varchar(25)	Usually the name of a country
SensorFamily	SensorID	Decimal(3)	SensorID will be a number between 1 - 500
SensorFamily	Is_DB110	bit	True or False to indicate if a certain family
SensorFamily	Is_MS110	bit	True or False to indicate if a certain family
SensorFamily	Is_MS177	bit	True or False to indicate if a certain family
SensorFamily	Is_TACSAR	bit	True or False to indicate if a certain family
IntegrationTest	SensorID		SensorID will be a number between 1 - 500
IntegrationTest	Test	Decimal(2)	1 = Controls, 2 = LOS, 3 = Thermal, 4 = Auxillary, 5 = Imaging
IntegrationTest	SubtestID	Decimal(1,1)	Numerical representation of a subtest such as '3.2' or '2.2'
IntegrationTest	Completion_Percentage	Decimal(3)	Up to 100%
IntegrationTest	DataLocation	Varchar(255)	Expect this to be a URL to a sharepoint location
Subtest	TestID	Decimal(1)	Numerica representation of Test 1 - Test 5
Subtest	SubtestID	Decimal(1,1)	Numerical representation of a subtest such as '3.2' or '2.2'
Subtest	SensorID	Decimal(3)	SensorID will be a number between 1 - 500
Subtest	Status	bit	0 = Fail, 1 = Pass



4.1 Normalization

Error Log Normalization:
PK : SensorID
FK: SensorID
Candidate Keys: All

1NF - PK Identified, no repeating groups
2NF - No Attributes dependent on the PK
3NF - No Attributes dependent on other attributes

ErrorLog

SensorID
FileName
RMSVersion
SCUVersion
GEDIVersion
AIBVersion

Integration Normalization
PK : SensorID
FK: SensorID
Candidate Keys: All

1NF - PK Identified, no repeating groups
2NF - No Attributes dependent on the PK
3NF - No Attributes dependent on other attributes

Integration Tests
SensorID
Test
SubTest
Completion%
DataLocation

Boot Table Normalization:
PK : SensorID
FK: SensorID
Candidate Keys: All

1NF - PK Identified, no repeating groups
2NF - No Attributes dependent on the PK
3NF - No Attributes dependent on other attributes

Boot
SensorID
Time
FileName
Message

SBIT Table Normalization:
PK : SensorID
FK: SensorID
Candidate Keys: All

- **1NF - PK Identified, no repeating groups**
- **2NF - No Attributes dependent on the PK**
- **3NF - No Attributes dependent on other attributes**

SBIT
SensorID
Time
FileName
Test
Status
Message

Plan Table Normalization:
PK : SensorID
FK: SensorID
Candidate Keys: All

- **1NF - PK Identified, no repeating groups**
- **2NF - No Attributes dependent on the PK**
- **3NF - No Attributes dependent on other attributes**

Plan
SensorID
Time
FileName
Tasks
Datalink
Task
PlanID
Message

Configuration File Table Normalization
PK : SensorID
FK: SensorID
Candidate Keys: All

1NF - PK Identified, no repeating groups
2NF - No Attributes dependent on the PK
3NF - No Attributes dependent on other attributes

ConfigurationFile
SensorID
RollResDevicepub.offset
RollResConfig.offset
PitchRes.Devicepub.offset
PitchResConfig.offset
BS.nf1bm0
BS.nf1bm1
BS.nf1bm2
BS.df1bm0
BS.df1bm1
BS.df1bm2

Sensor Family Table Normalization:
PK : SensorID
FK: SensorID
Candidate Keys: All

1NF - PK Identified, no repeating groups
2NF - No Attributes dependent on the PK
3NF - No Attributes dependent on other attributes

SensorFamily
SensorID
Is_DB110
Is_MS110
Is_MS177

Subtest Table Normalization:
PK : SensorID
FK: SensorID
Candidate Keys: All

1NF - PK Identified, no repeating groups
2NF - No Attributes dependent on the PK
3NF - No Attributes dependent on other attributes

Subtests
TestID
SubtestID
SensorID
Status

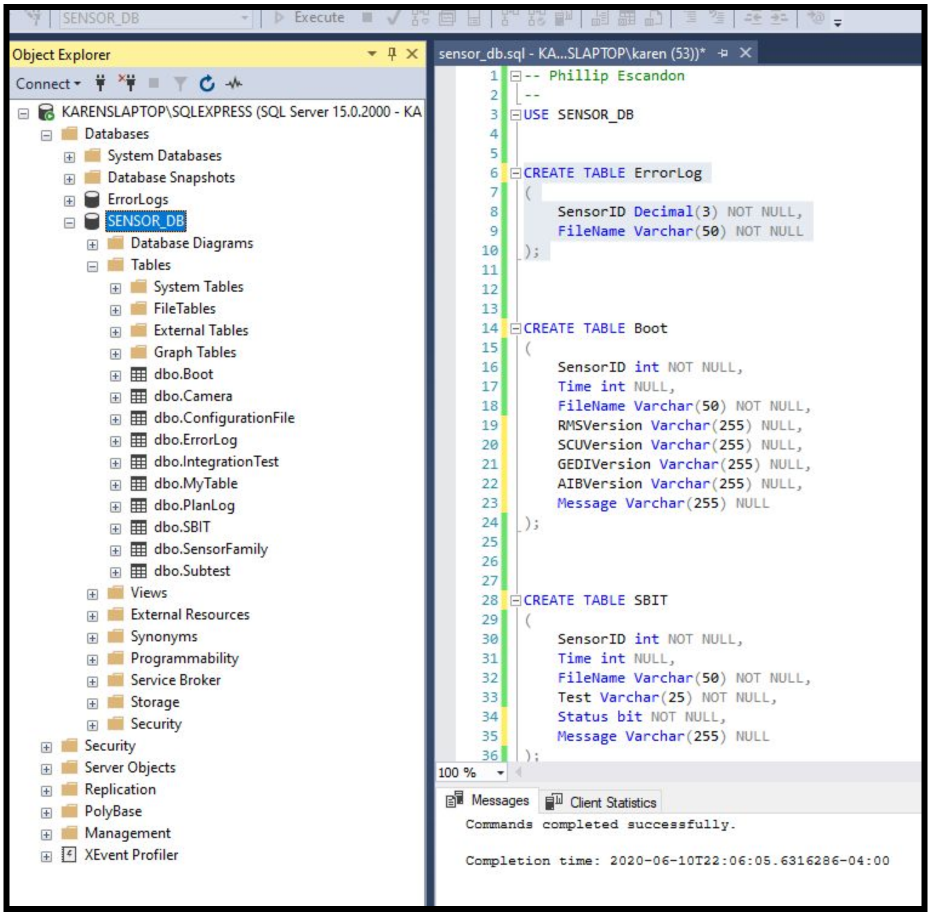
Camera Table Normalization:
PK : SensorID
FK: SensorID
Candidate Keys: All

1NF - PK Identified, no repeating groups
2NF - No Attributes dependent on the PK
3NF - No Attributes dependent on other attributes

Camera
SensorID
Customer
RollBench
PitchBench

4.2 Tables and Constraints

See attached SQL File.



4.3 Index Placement and Creation

Primary Keys:

Primary Keys	Description
ErrorLog.SensorID	They key attribute that threads the entire database is the SensorID. Everything else is secondary.
Boot.SensorID	
SBIT.SensorID	
PlanLog.SensorID	

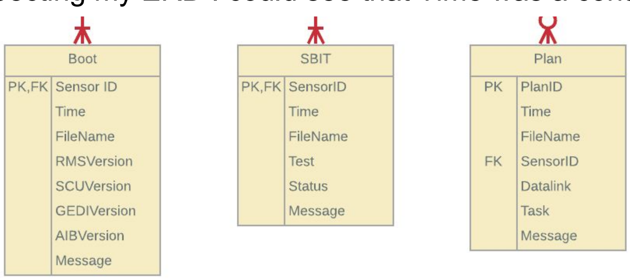
ConfigurationFile.SensorID	
Camera.SensorID	
SensorFamily.SensorID	
IntegrationTest.SensorID	
Subtest.SensorID	

Foreign Keys:

Foreign Keys	
ErrorLog.SensorID	This FK reference the Camera table.The index is non-unique since many errorlog can be in the same camera.
PlanLog.SensorID	This FK reference the Camera table.The index is non-unique since many errorlog can be in the same camera.
Subtests.SensorID	This FK reference the Camera table.The index is non-unique since many errorlogs can be in the same camera.
Subtests.TestID	This FK reference the Camera table.The index is non-unique since many errorlogs can be in the same camera.

Use Cases:

Inspecting my ERD I could see that Time was a consistent attribute across the logs.

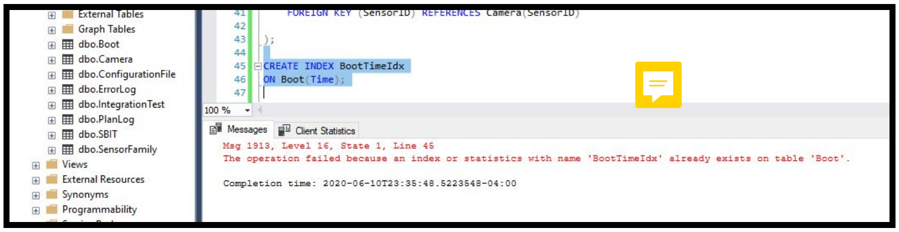


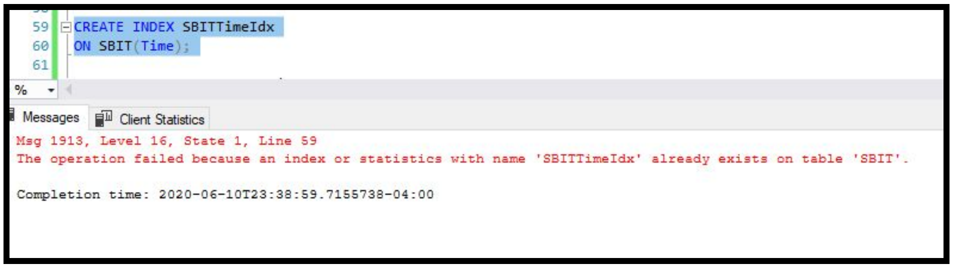
Use Case:

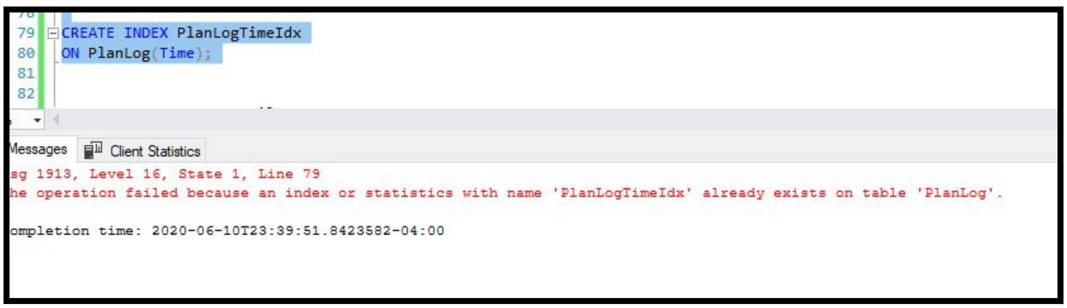
An Engineer wanted to inspect all logs that occurred on two specific days.

Pseudocode:

Select SBIT Messages and Plan IDs WHERE TIME > (some date).



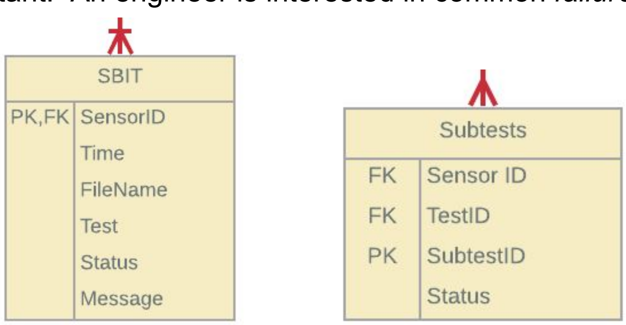




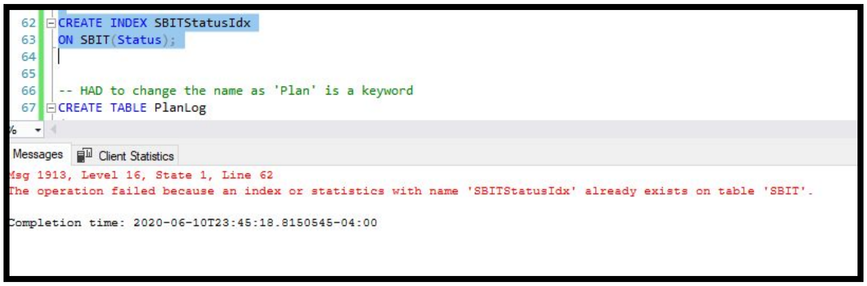
The entire script was executed - on re-execution it was verified that the index was already created.

Use Case 2:

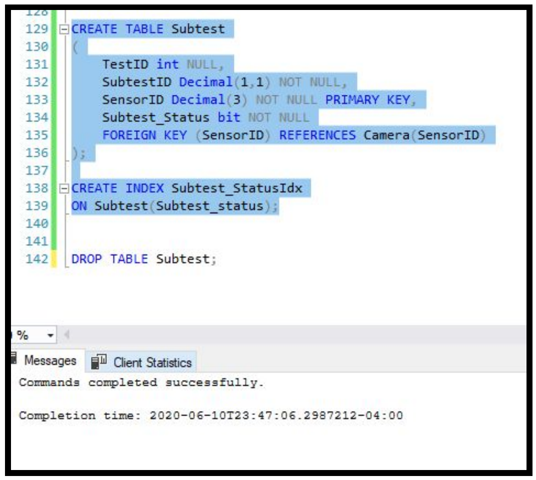
The history of test status is important. An engineer is interested in common *failures between* SBIT and Subtests.



The Status attribute is a possible candidate for indexing.



Re-executing the indexing gave me an error stating that it was already indexed when the script was executed.



5. Reusable Transaction - Oriented Store Procedures

5.1 History Table

5.2 Questions and Queries