AWS Data, Databases, and Analytics Online Series

AWS 데이터레이크 환경에서의 데이터 분석 플랫폼 구축 사례

정현아

Solutions Architect (emmajung@amazon.com)

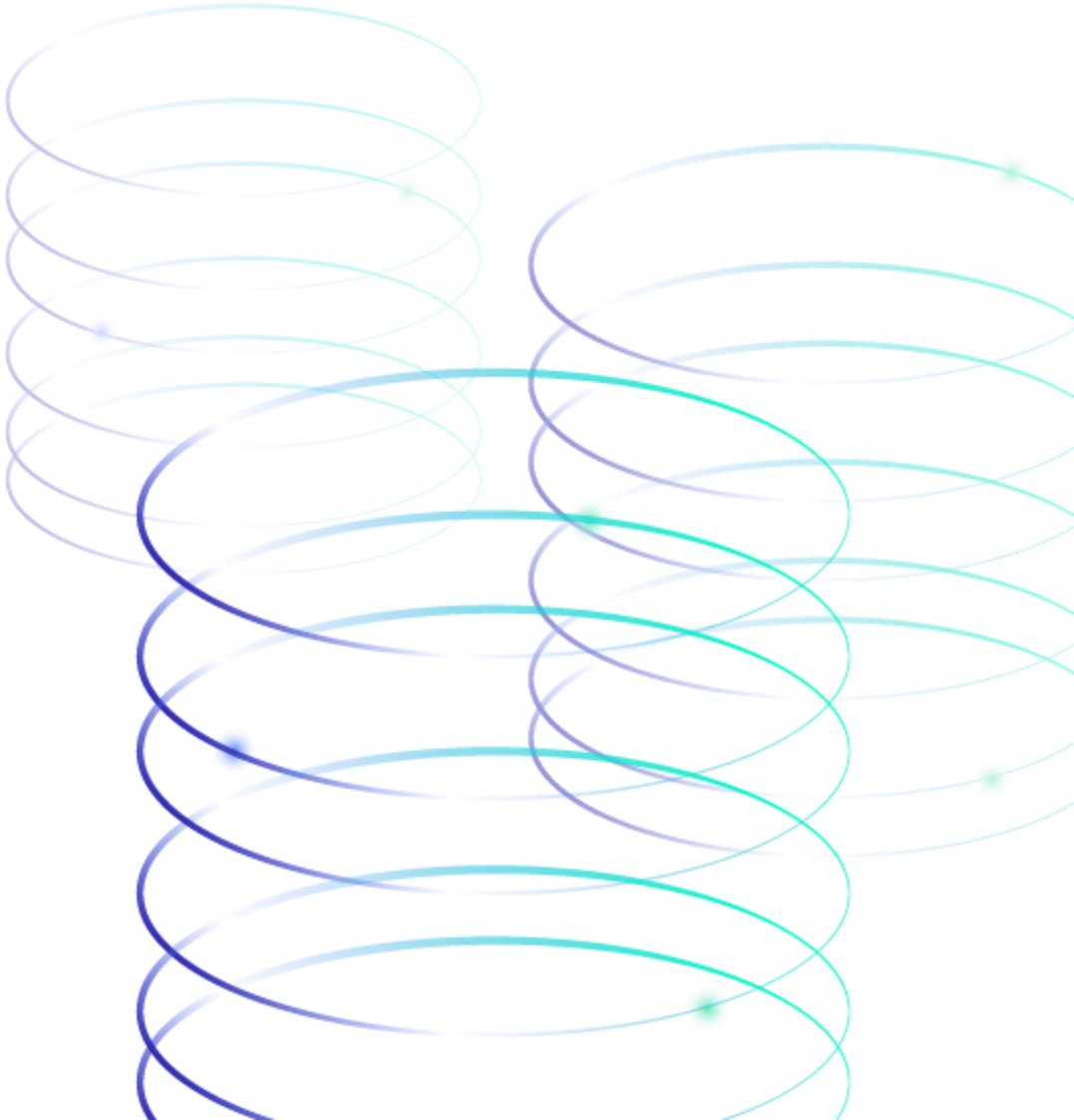
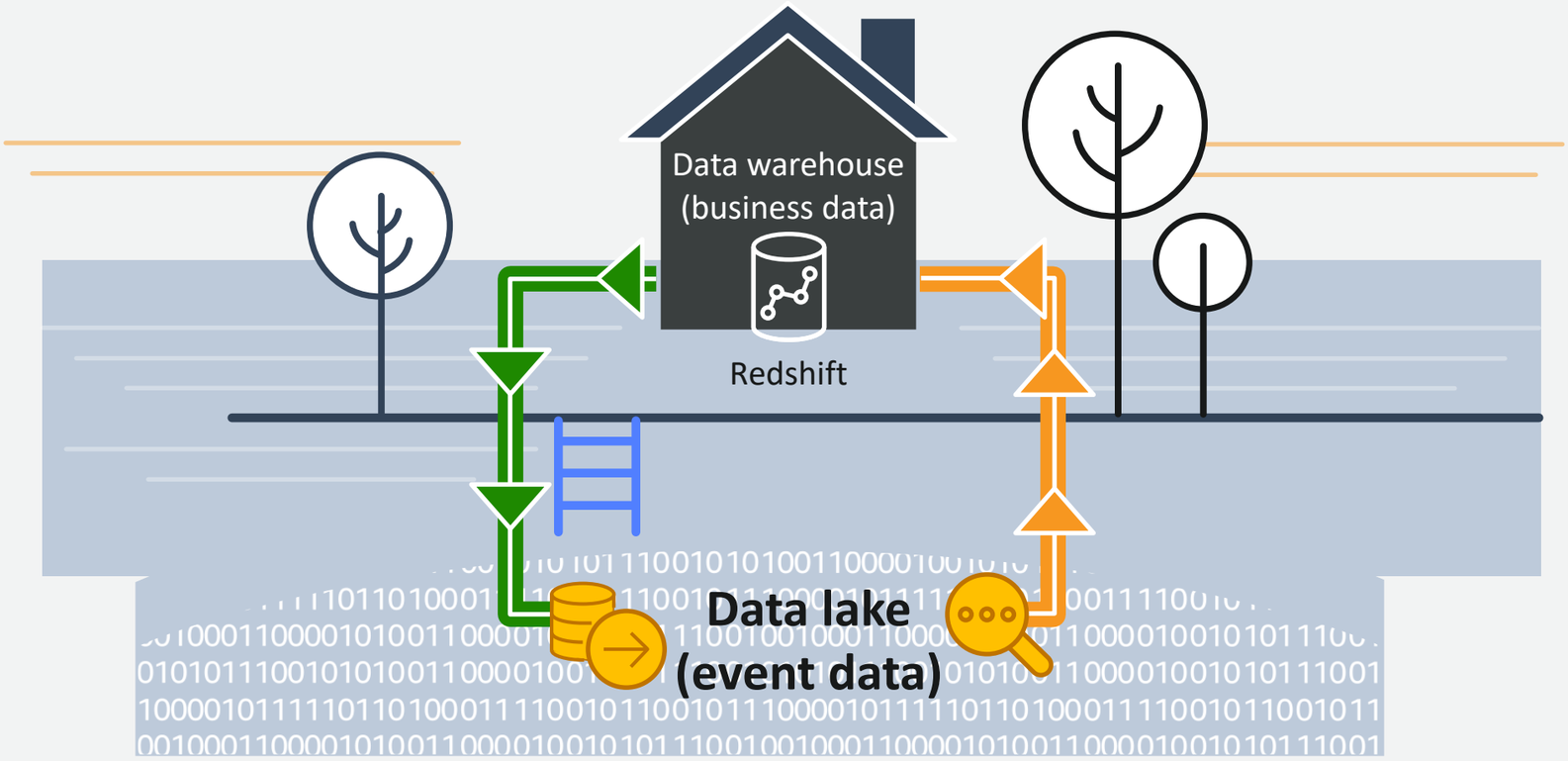


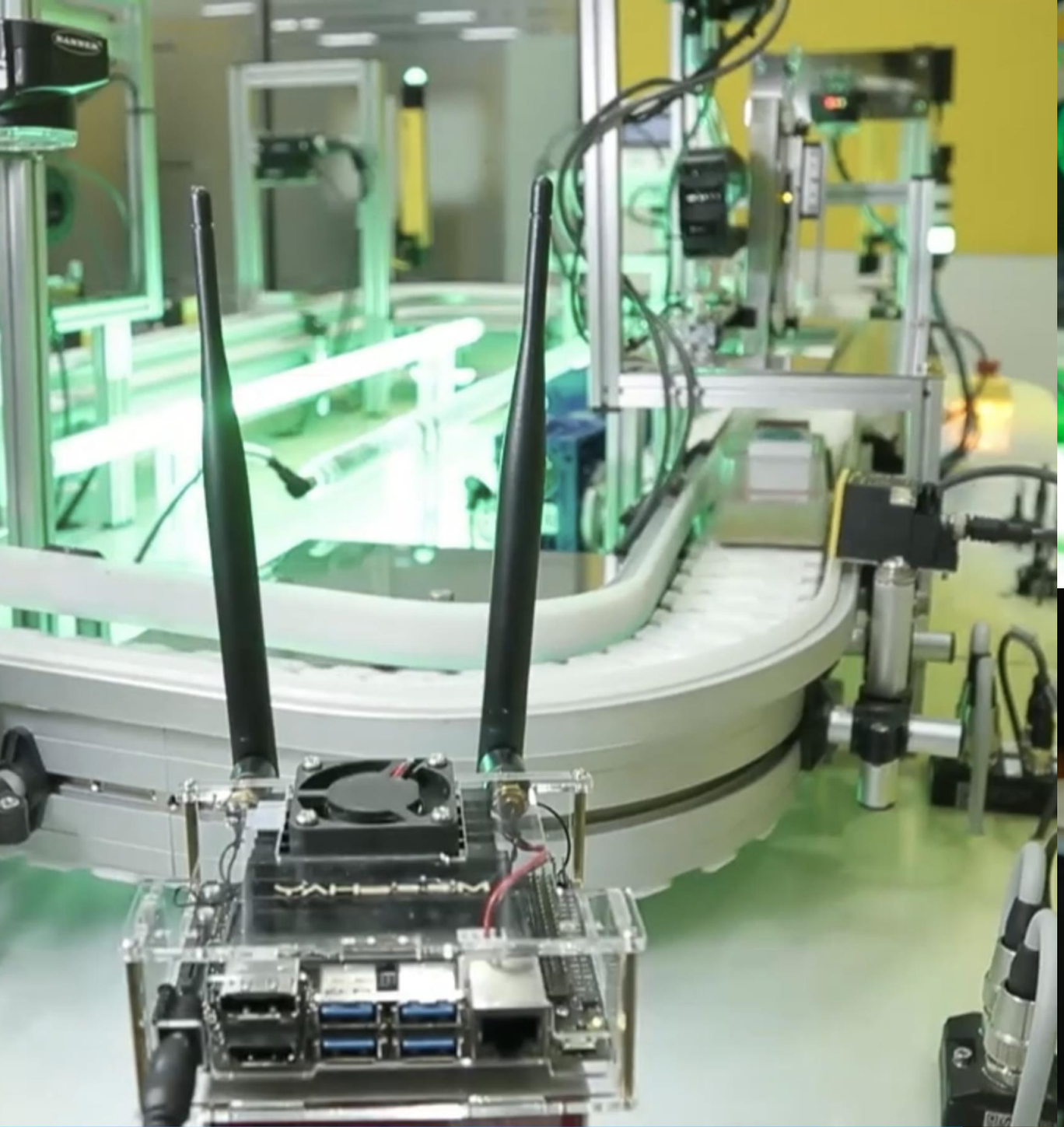
Table of contents

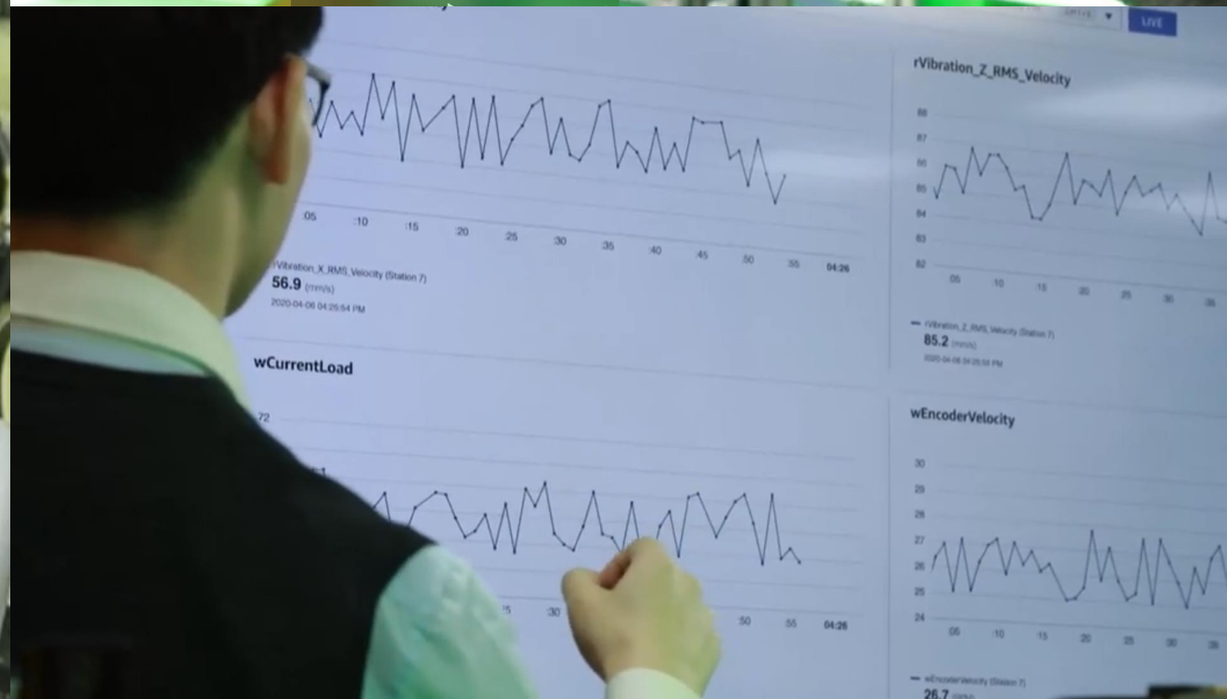
Data lake: The new information hub



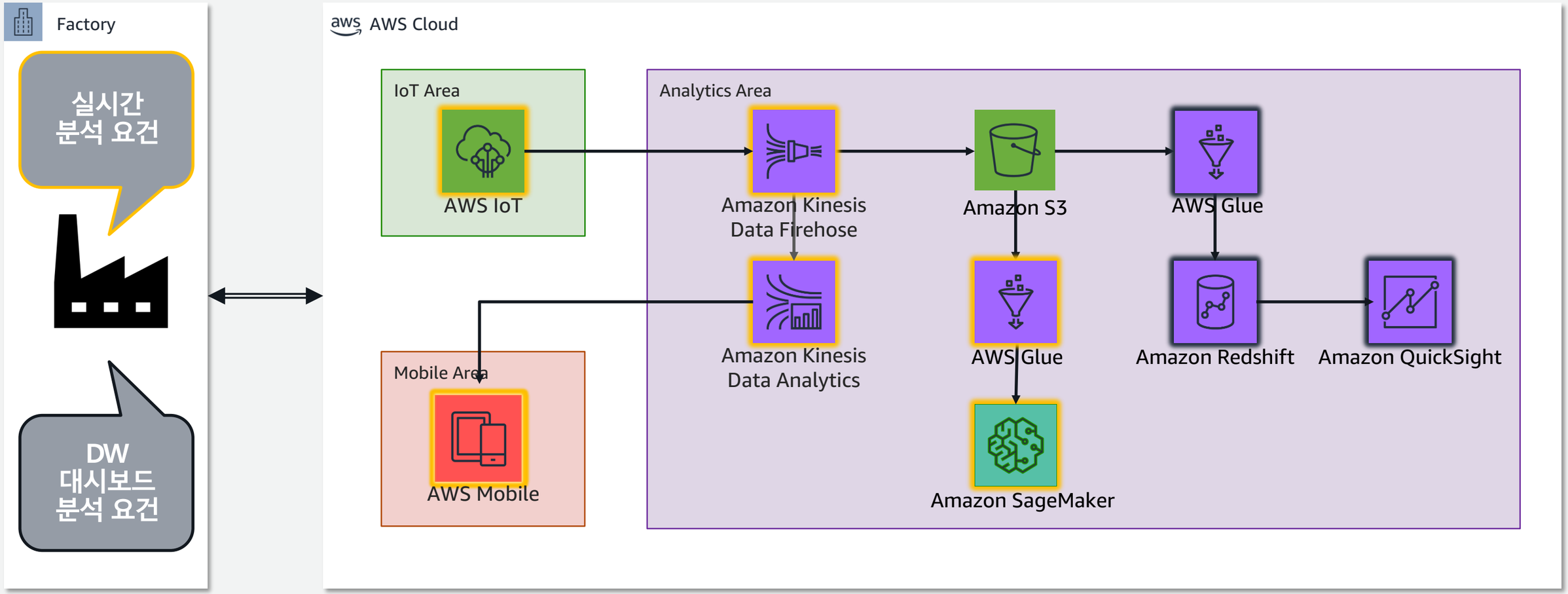
고객들은 Data Lake 아키텍처로 이동 중

Redshift가 Data Lake House 접근법을 가능하게 해줌




LIVE
rVibration_Z_RMS_Velocity
56.9
85.2
wCurrentLoad
wEncoderVelocity

Smart Factory Architecture



핵심 구현 요소 – 실시간 vs. 배치성

실시간성 분석

실시간 모니터링 및 장애 확률 감지

- 로그/센서 데이터 등의 공정데이터
- Kinesis 서비스를 통해 DataLake로 초 단위 지속적 데이터 수집 및 실시간 분석 수행
- 과거 공정데이터 바탕으로 Sagemaker ML 모델 빌드

DataLake

DataLake로 모든 분석 기반 데이터 수집

- 실시간/배치 레이어를 구분
- 각 레이어 별 대상 데이터를 별도로 저장
- 레이어 분리를 통한 리소스 경합 최소화

배치성 분석

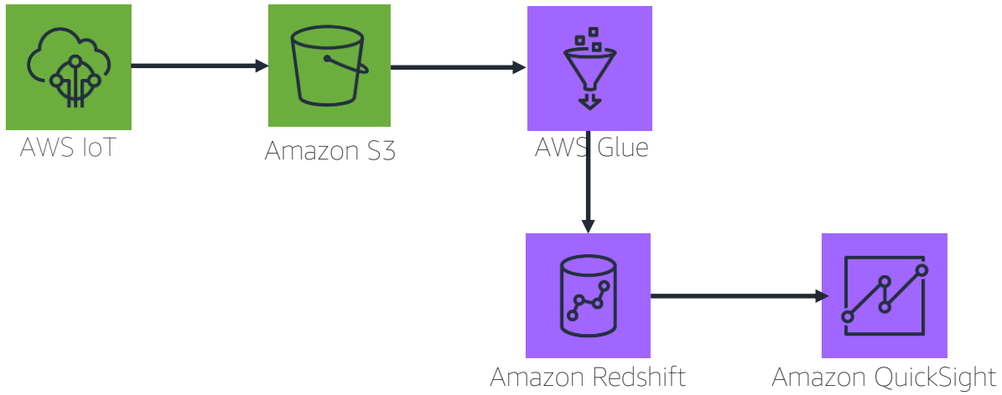
DW 분석 및 시각화 대시보드 구성

- 로그/센서 데이터 등의 공정 데이터 + ERP/리테일 생산데이터
- Glue서비스를 통해 DataLake에 수집된 데이터 중 일별 배치로 필요한 데이터를 Redshift로 ETL
- 실시간 데이터 S3에서 바로 Spectrum 서비스 사용

DataWarehouse 대시보드 분석

DW 대시보드 분석 요건 및 아키텍쳐

DW 구축 및 BI 대시보드 구축

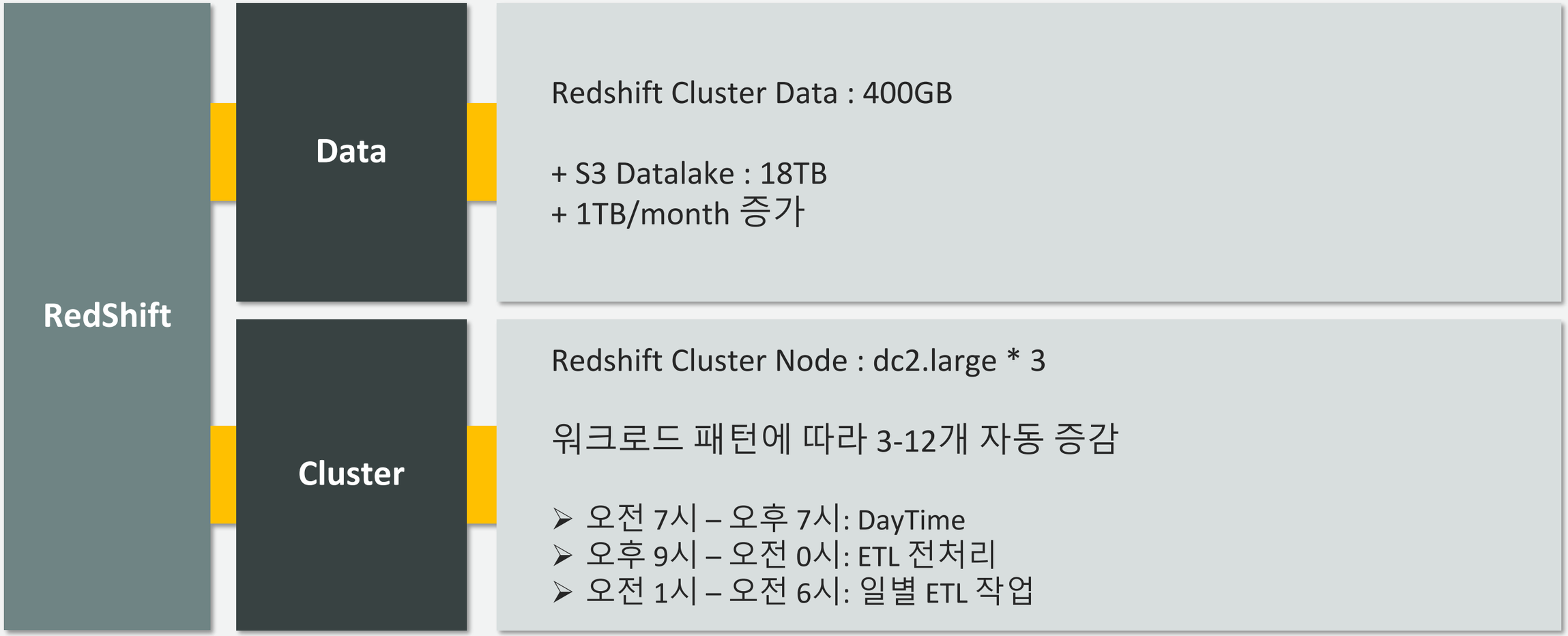


비즈니스 현황 확인 및 공정 데이터 분석 위한

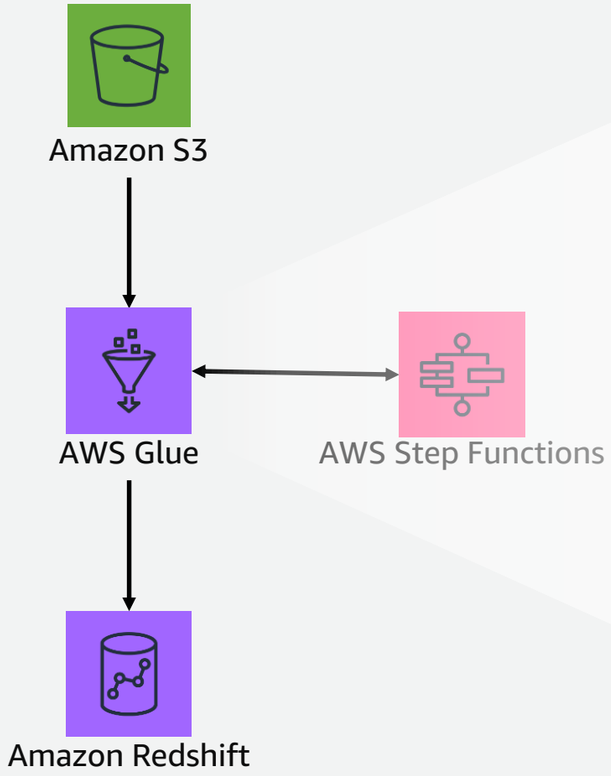
DW구축 및 BI 대시 보드 구축

- 히스토리, 실시간 공정데이터, 생산데이터 S3에 Datalake 구축
  - ✓ 실시간 수집 + 일배치 수집
- Glue + StepFunction 로 단계별 ETL 작업 플로우 구성
- Redshift로 데이터웨어 하우스 구성
- Quick Sight 로 시각화 대시보드 구성

스마트팩토리 on Redshift



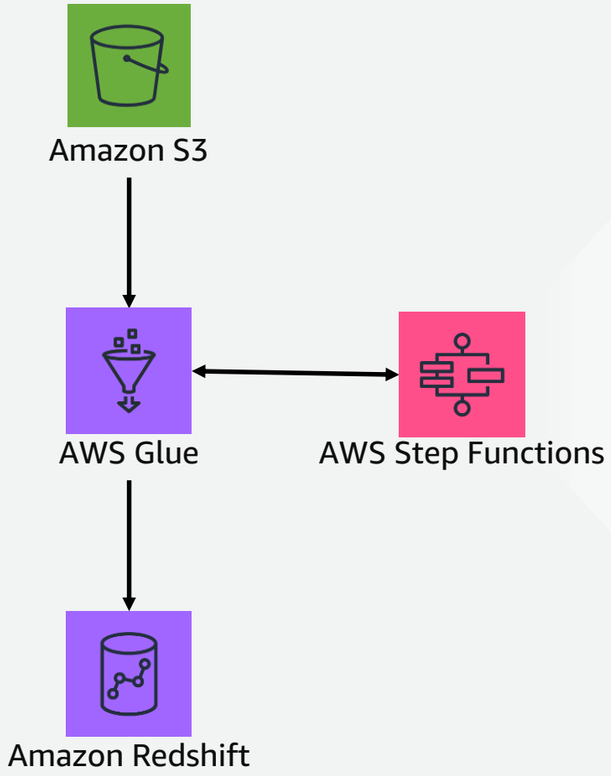
DW 대시보드 분석 요건 – Glue + Step Function

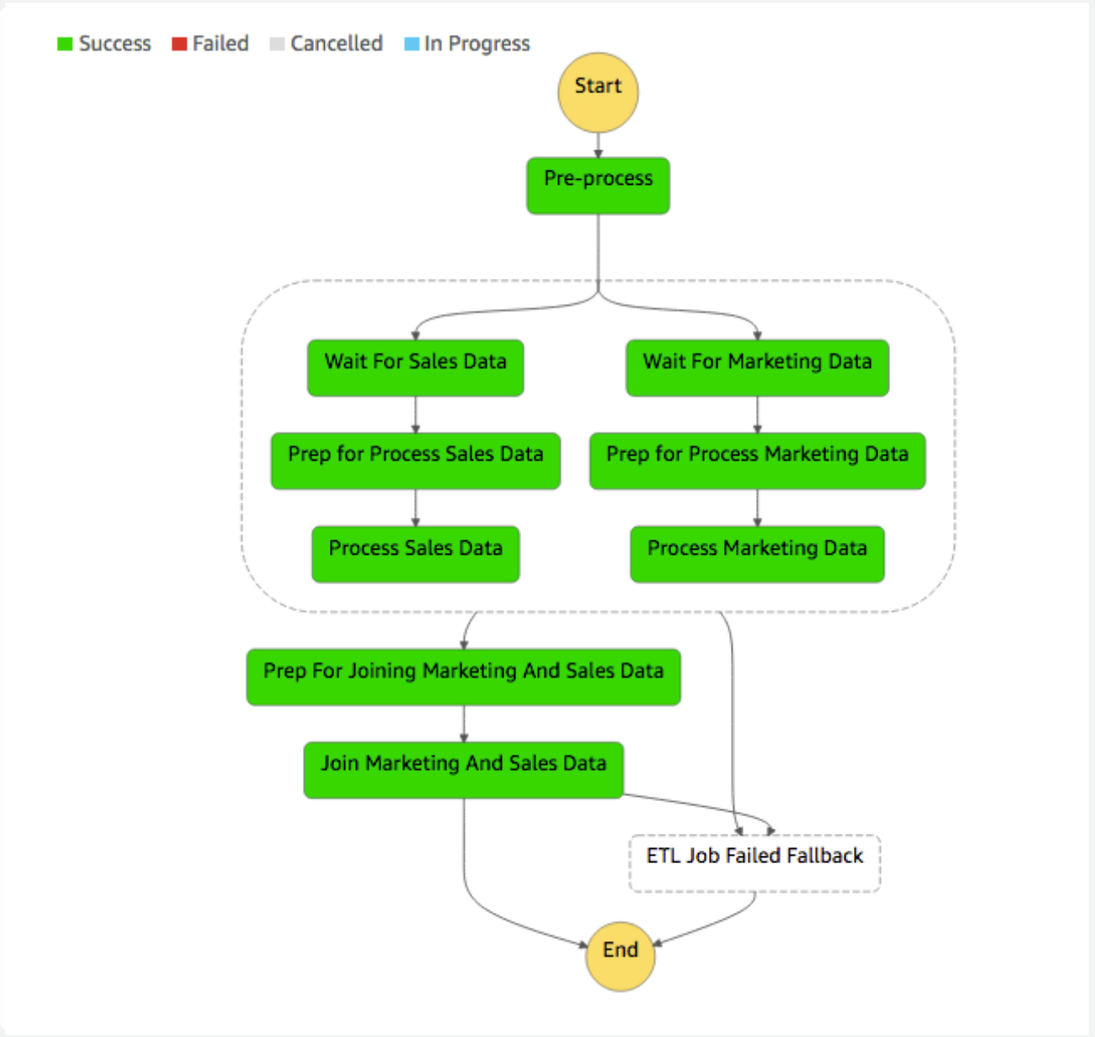


Job: ETL_Daily_Fact_Retail_sales Action Save Run job

```python
import pg
import boto3
import base64
from botocore.exceptions import ClientError
import json

#uses session manager name to return connection and credential information
def connection_info(db):

    session = boto3.session.Session()
    client = session.client(
        service_name='secretsmanager'
    )

    get_secret_value_response = client.get_secret_value(SecretId=db)

    if 'SecretString' in get_secret_value_response:
        secret = json.loads(get_secret_value_response['SecretString'])
    else:
        secret = json.loads(base64.b64decode(get_secret_value_response['SecretBinary']))

    return secret
```

DW 대시보드 분석 요건 – Glue + Step Function





DW 대시보드 분석 요건 – Redshift Spectrum

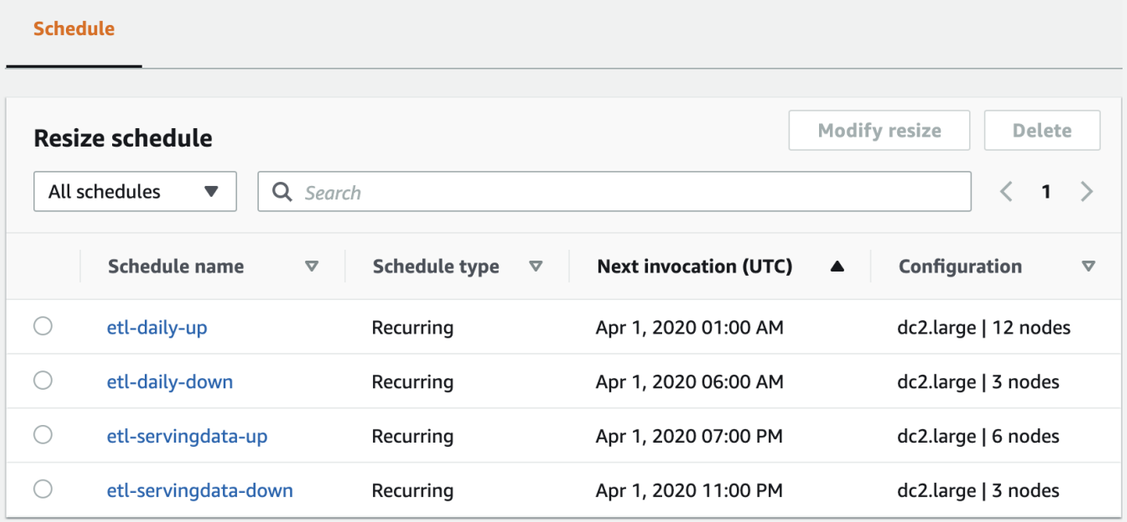
- 데이터레이크와의 결합
  - ✓ 6개월 이전의 FACT 테이블 데이터는 S3에 보관
  - ✓ 스키마 바인딩 뷰로 내/외부 테이블 지정
    - 데이터 저장 비용 최적화
    - 대부분의 BI Report는 Summary 테이블 및 뷰를 참조

aws

DW 대시보드 분석 요건 – Redshift Spectrum

- 데이터레이크와의 결합
  - ✓ 6개월 이전의 FACT 테이블 데이터는 S3에 보관
  - ✓ 스키마 바인딩 뷰로 내/외부 테이블 지정
    - 데이터 저장 비용 최적화
    - 대부분의 BI Report는 Summary 테이블 및 뷰를 참조
- 성능 개선 포인트
  - ✓ 대용량 Fact 테이블 간의 여러 단계의 조인은 지양
    - Staging 테이블을 활용하여 Spectrum fleet으로 Pushdown
  - ✓ ETL 및 Report 참조 시 구체화된 뷰의 미리 계산된 데이터를 활용
    - RA3.4xlarge 노드로 변경 예정

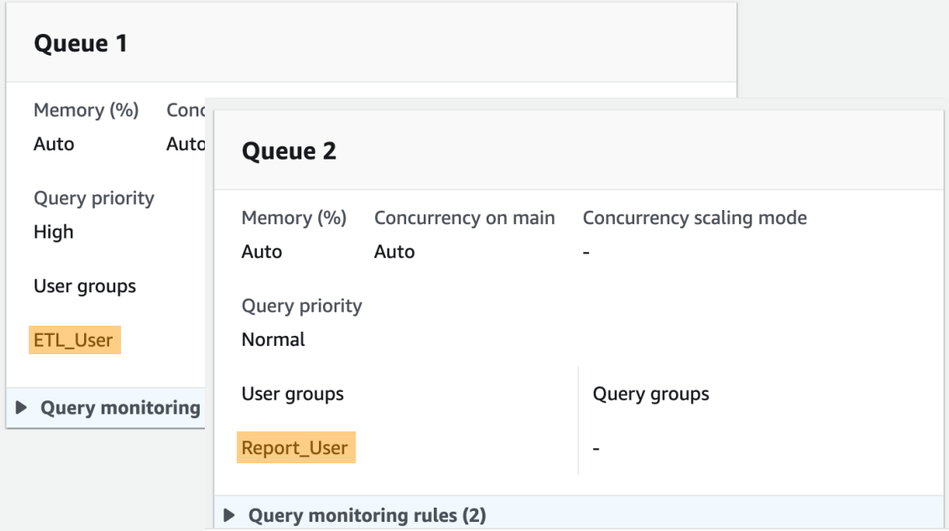
DW 대시보드 분석 요건 – Scaling

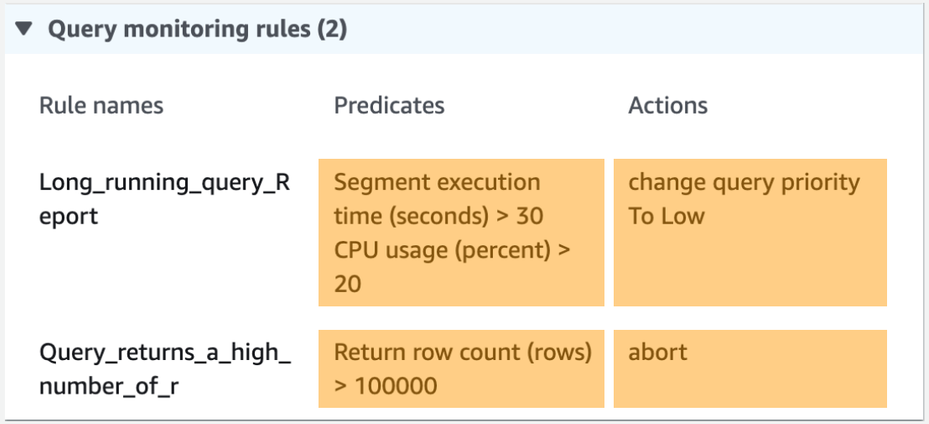


- **Scheduled Elastic Resizing**
  - ✓ 일별 ETL 실행 시간 (01시 – 06시): dc2.8xlarge * 12
  - ✓ 업무 시간 (06시 – 19시) : dc2.8xlarge * 3
  - ✓ 데이터 서빙 시간 (19시 – 23시) : dc2.8xlarge * 6

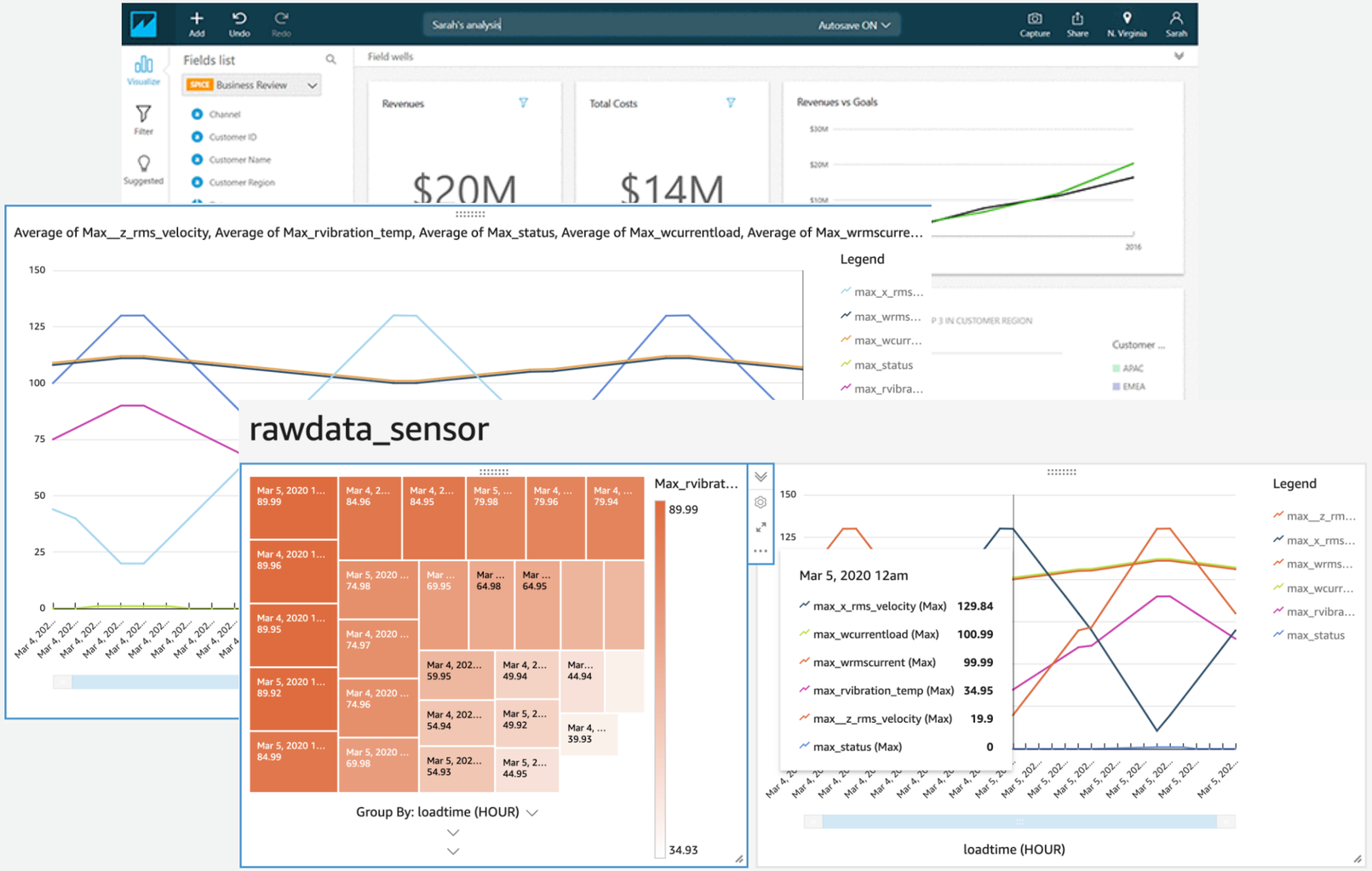
DW 대시보드 분석 요건 – Scaling

- Auto WLM(워크로드 관리) + QMR
  - ✓ 워크로드 별 사용자 그룹 및 쿼리 대기열 생성
    - ETL_User
    - BI_Report_User
    - Analytics_User
  - ✓ WLM 모드는 Auto로 지정(기본값)
  - ✓ 워크로드에 따라 자동으로 리소스가 할당되도록 설정





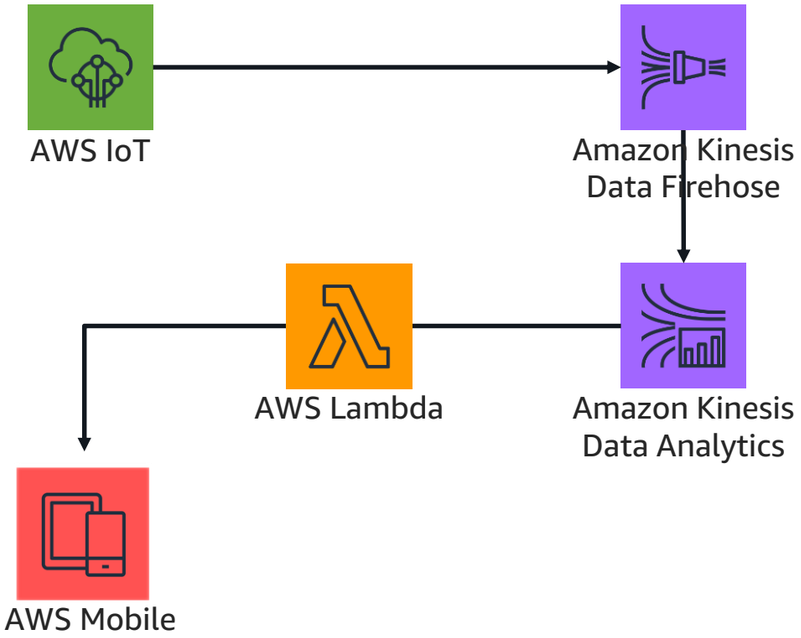
DW 대시보드 분석 요건 – 시각화 대시보드



실시간 분석플랫폼

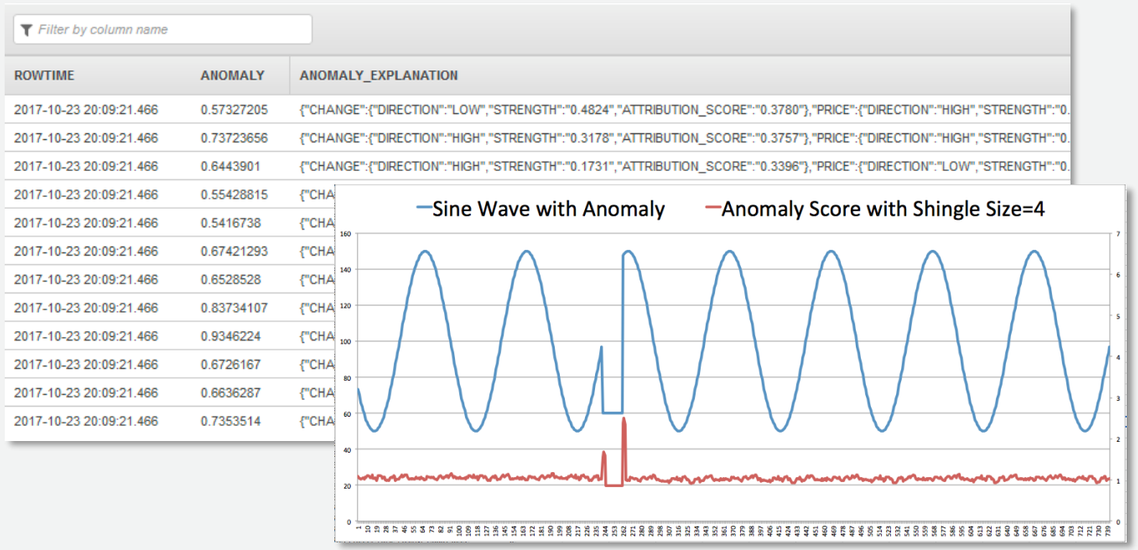
실시간 분석 요건 - 실시간 이상치 감지 알림

실시간 이상치 감지 알림

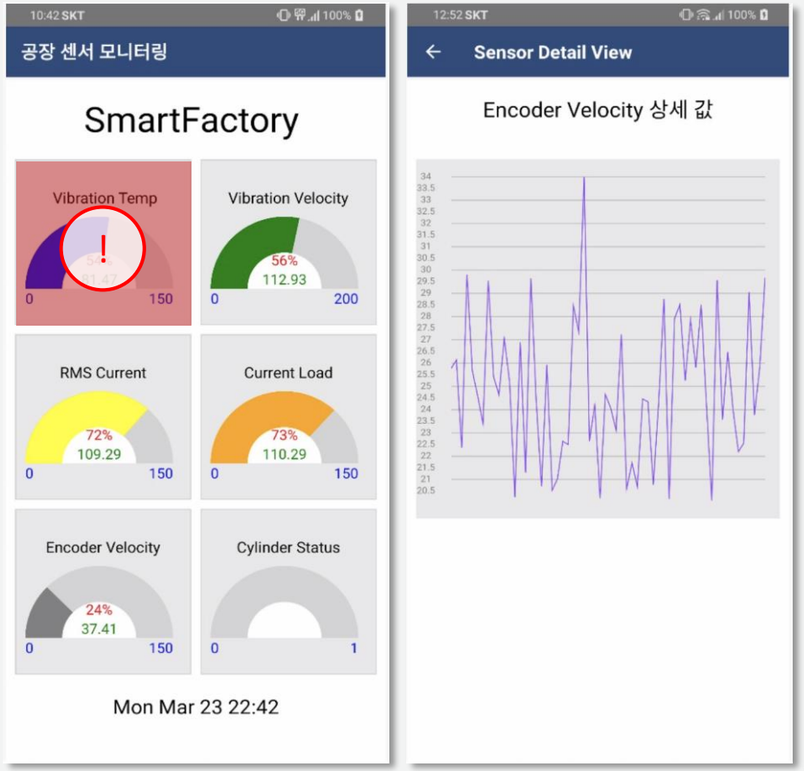


- Kinesis 서비스를 통해 IoT Core에서 센서데이터 실시간 스트리밍
- Kinesis Data Analytics 서비스에서 데이터 변칙 발생 시 이를 실시간으로 탐지
  - ✓ Random Forest 알고리즘 사용하여 센서값 이상 정도에 따른 score 추출, 임계치 이상의 Score 발생 필터링 알림
  - ✓ 3분마다 온도 10초 이동 평균 계산하여 결과 값 전달
- 사용자는 Mobile Application을 통해 이상치 발생 시 push 알림 수신

실시간 이상치 감지 알림



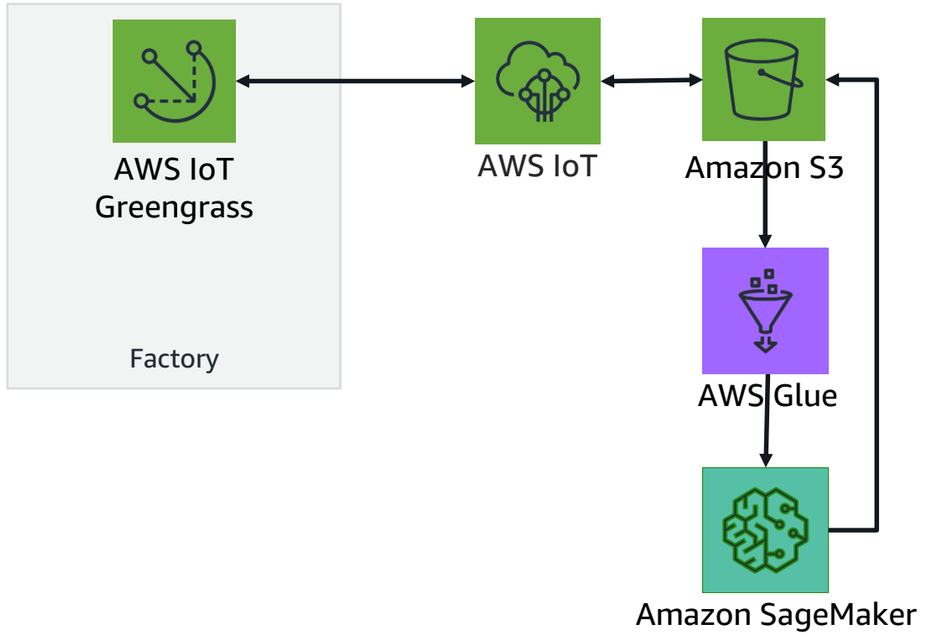
Kinesis Analytics



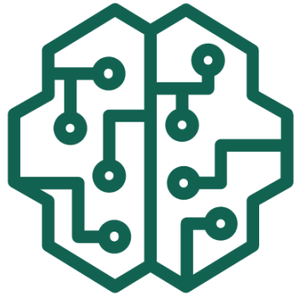
Mobile Application

실시간 분석 요건 – 예지 정비 모델 구현

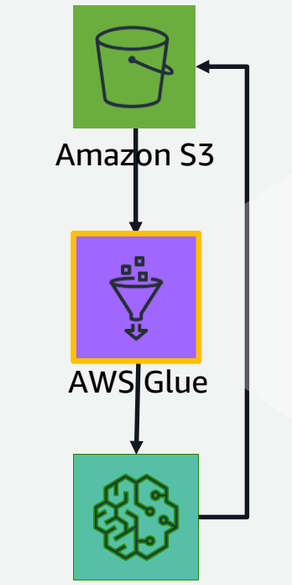
예지 정비 모델 구현

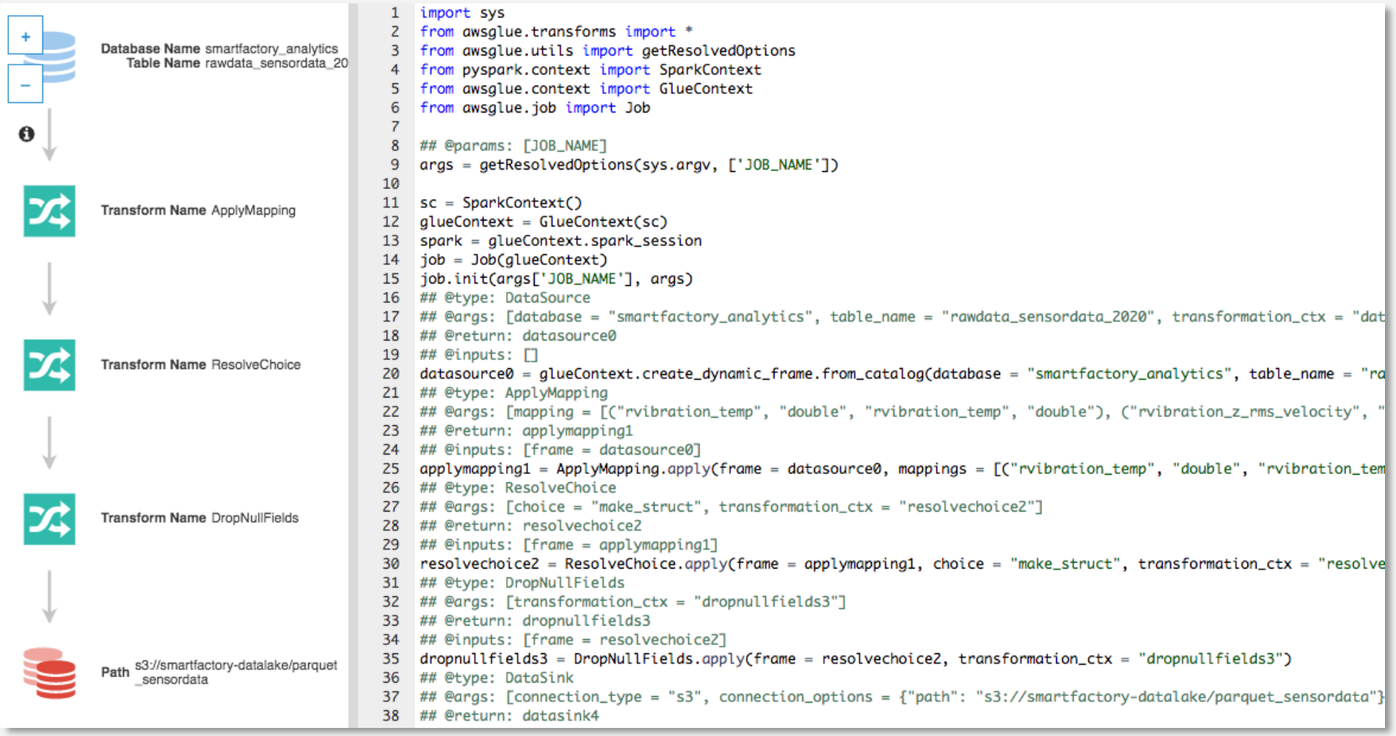


현재 온도, 모터속도, 진동 등 센서데이터의
수치로 보았을때
실린더가 고장날 확률은 86%입니다.



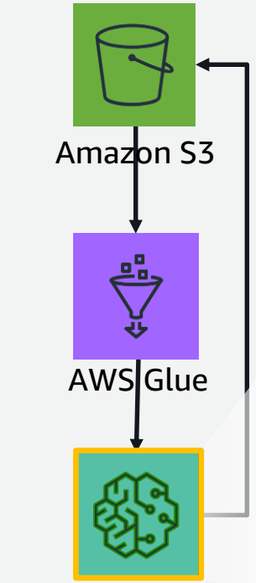
예지 정비 모델 구현 – 데이터 변환






```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)
## @type: DataSource
## @args: [database = "smartfactory_analytics", table_name = "rawdata_sensordata_2020", transformation_ctx = "dat
## @return: datasource0
## @inputs: []
datasource0 = glueContext.create_dynamic_frame.from_catalog(database = "smartfactory_analytics", table_name = "ra
## @type: ApplyMapping
## @args: [mapping = [("rvibration_temp", "double", "rvibration_temp", "double"), ("rvibration_z_rms_velocity", "
## @return: applymapping1
## @inputs: [frame = datasource0]
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [("rvibration_temp", "double", "rvibration_tem
## @type: ResolveChoice
## @args: [choice = "make_struct", transformation_ctx = "resolvechoice2"]
## @return: resolvechoice2
## @inputs: [frame = applymapping1]
resolvechoice2 = ResolveChoice.apply(frame = applymapping1, choice = "make_struct", transformation_ctx = "resolve
## @type: DropNullFields
## @args: [transformation_ctx = "dropnullfields3"]
## @return: dropnullfields3
## @inputs: [frame = resolvechoice2]
dropnullfields3 = DropNullFields.apply(frame = resolvechoice2, transformation_ctx = "dropnullfields3")
## @type: DataSink
## @args: [connection_type = "s3", connection_options = {"path": "s3://smartfactory-datalake/parquet_sensordata"}
## @return: datasink4
```

예지 정비 모델 구현 – ML 모델



jupyter smartfactory_xgboost_2020-03-06 Last Checkpoint: 2020.03.11 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

```
                                    role,
                                    train_instance_count=1,
                                    train_instance_type='ml.m4.xlarge',
                                    output_path='s3://{}/{}/output'.format(bucket, prefix),
                                    sagemaker_session=sess)
```

In [18]:

```
#Parameter setting
xgb.set_hyperparameters(
        objective = 'binary:logistic',
        booster = "gbtree",
        eval_metric = 'auc',
        scale_pos_weight = 10,
        nthread = 4,
        eta = 0.05,
        max_depth = 5,
        subsample = 0.7,
        colsample_bytree = 0.7,
        random_state = 42,
        nrounds = 200,
        n_estimators = 200,
        num_round=25
    )
```

In [19]:

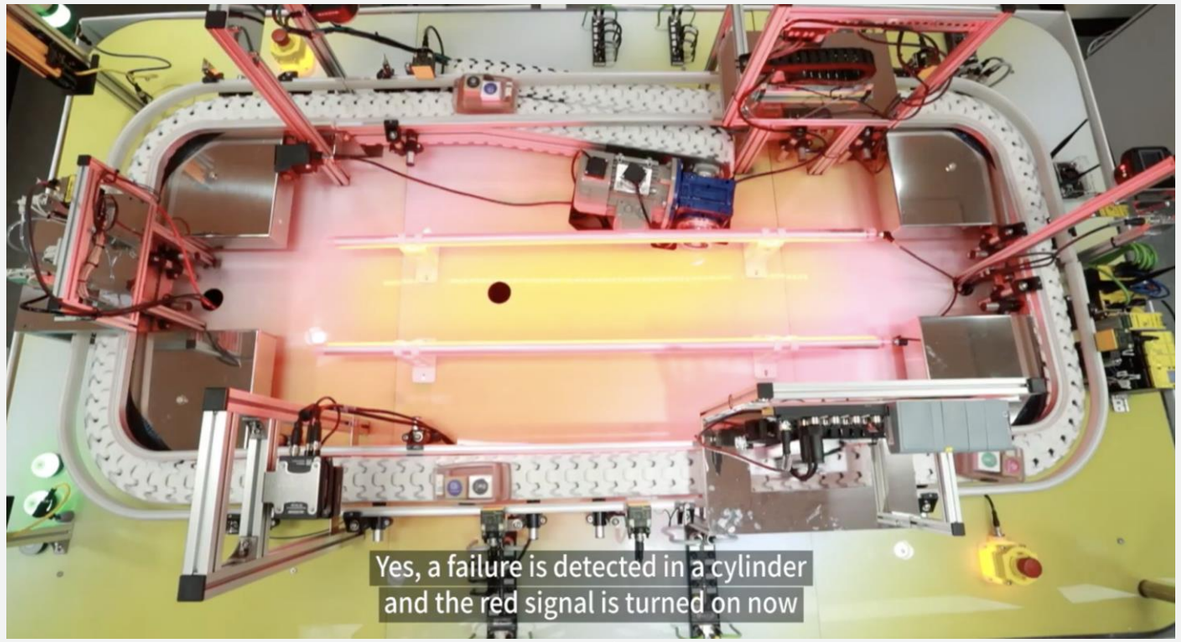
```
#training model
xgb.fit({'train': s3_input_train, 'validation': s3_input_validation})
```

```
2020-03-06 00:49:03 Starting - Starting the training job...
2020-03-06 00:49:05 Starting - Launching requested ML instances......
2020-03-06 00:50:08 Starting - Preparing the instances for training......
2020-03-06 00:51:31 Downloading - Downloading input data
```

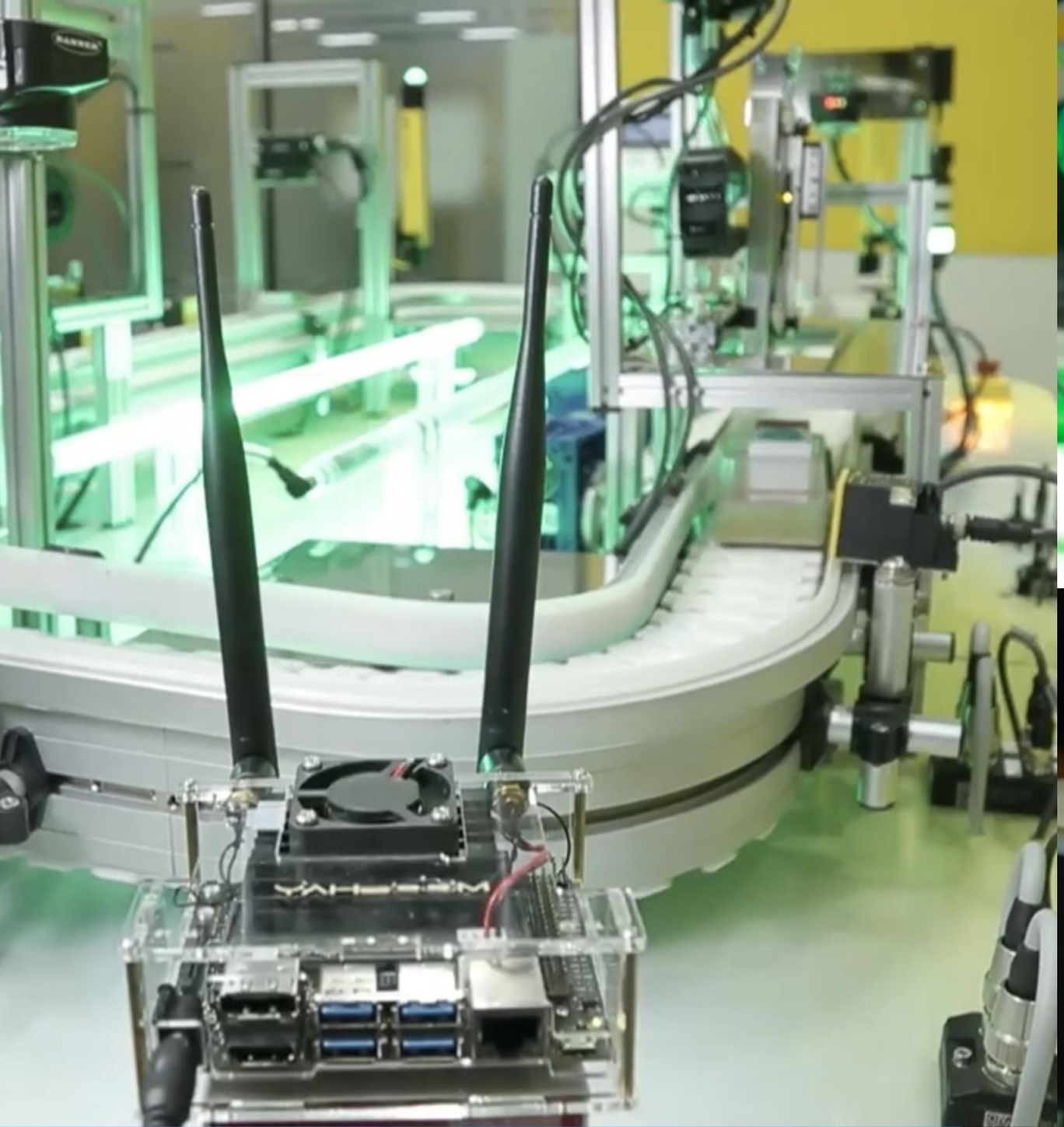
예지 정비 모델 구현 – Inference

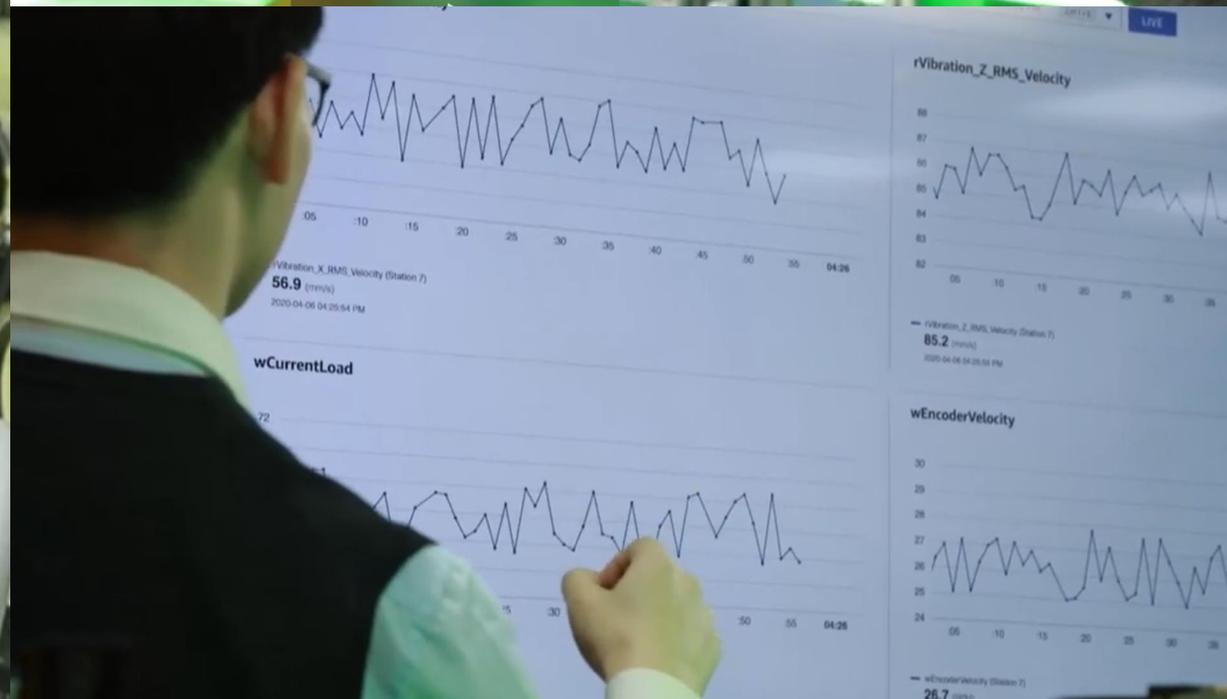
Sagemaker

```
# functions
#
def classify_data(test_input, model, threshold=0.5, context_type='application/json'):
    """
    Perform inference with the trained model.
    """
    test_input = np.expand_dims(test_input, axis=0) # need to expand 2-dimensional array
    str_test_input = str(test_input.tolist())  # convert to string
    parsed = json.loads(str_test_input)
    logger.debug("parsed: {}".format(parsed))

    dtest_input = xgb.DMatrix(parsed) # convert to xgb DMatrix type
    score = float(model.predict(dtest_input)[0]) # get prediction score
    pred = int(score > threshold) # get classification result
    logger.info("prediction: {}".format(pred))
    dic = {'score': score, 'pred': pred}
    response_body = json.dumps(dic)
    logger.info("response_body: {} output_content_type: {}".format(response_body, context_type))

    return response_body, context_type

def test_online(test_input, model):
    """
    Perform inference with one sample data as input for online(real-time) test.
    """
    response_body, context_type = classify_data(test_input, model)
    return response_body, context_type
```

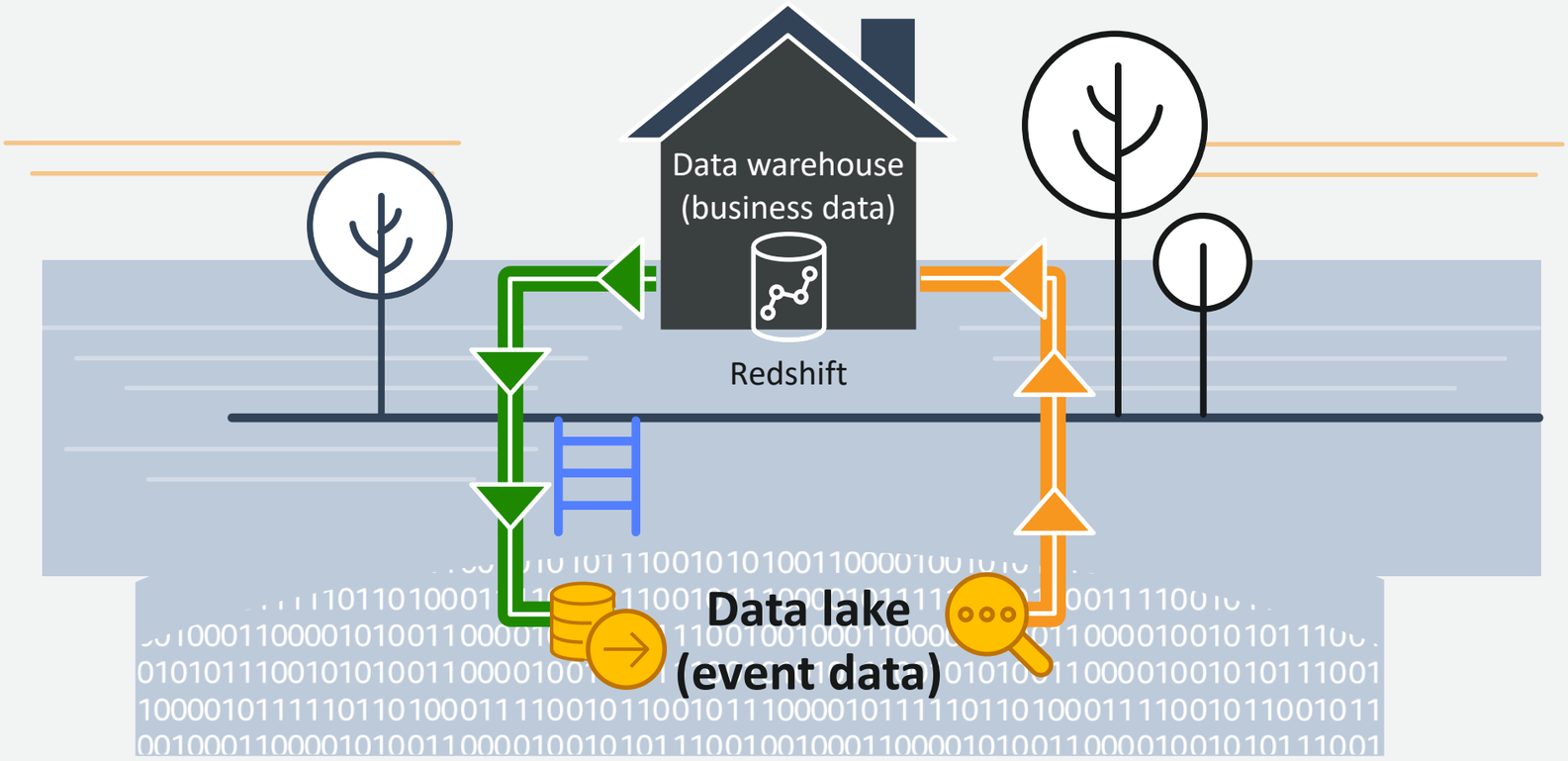


S3 Deploy → model.tar.gz → **Local Device Inferencing**




LIVE
rVibration_Z_RMS_Velocity
56.9
85.2
wCurrentLoad
wEncoderVelocity

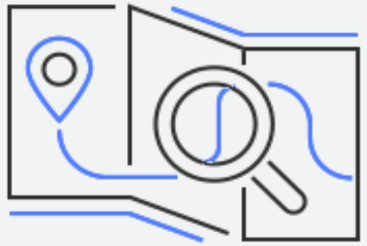
Data lake: The new information hub



고객들은 Data Lake 아키텍처로 이동 중

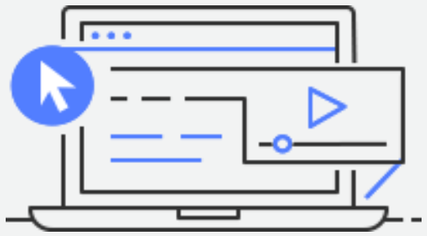
Redshift가 Data Lake House 접근법을 가능하게 해줌

AWS 교육 및 자격증



조직을 위한 맞춤 교육

고객 및 파트너를 위해 준비된 데이터 및 데이터베이스 관련 맞춤형 교육 여정을 확인해 보세요.



원하는 방법으로 - 유연한 학습 형태

"The elements of Data Science" 과정을 포함한 무료 디지털 교육 또는 강의실 교육을 통해 클라우드 역량을 향상시키세요.

AWS 자격증을 통한 기술 역량 입증

업계에서 인정받는 데이터 분석 또는 데이터베이스 – 전문분야 AWS 자격증을 통해 전문성을 입증할 수 있습니다.

aws.amazon.com/training

AWS 데이터분석 관련 자료를 원하시면 ...

데이터 분석 관련 기술 백서 및 전자책을 자세히 살펴보면
데이터에서 새로운 통찰력과 가치를 발견 할 수 있습니다!

- 클라우드 기반 데이터 분석 서비스
- 데이터의 분석 활용 사례
- 최신 분석 아키텍처 생성 방법
- 데이터 중심 기업 전환
- 동영상, 기술 백서 등

지금 방문하세요! »

https://tinyurl.com/data-databases-analytics-kr

AWS 데이터 분석 특집 웨비나에 참석해주셔서 대단히 감사합니다.

저희가 준비한 내용, 어떻게 보셨나요?
더 나은 세미나를 위하여 **설문을 꼭 작성해 주시기 바랍니다.**


aws-korea-marketing@amazon.com


twitter.com/AWSKorea


facebook.com/amazonwebservices.ko


youtube.com/user/AWSKorea


slideshare.net/awskorea


twitch.tv/aws

Thank you ☺