예측과정

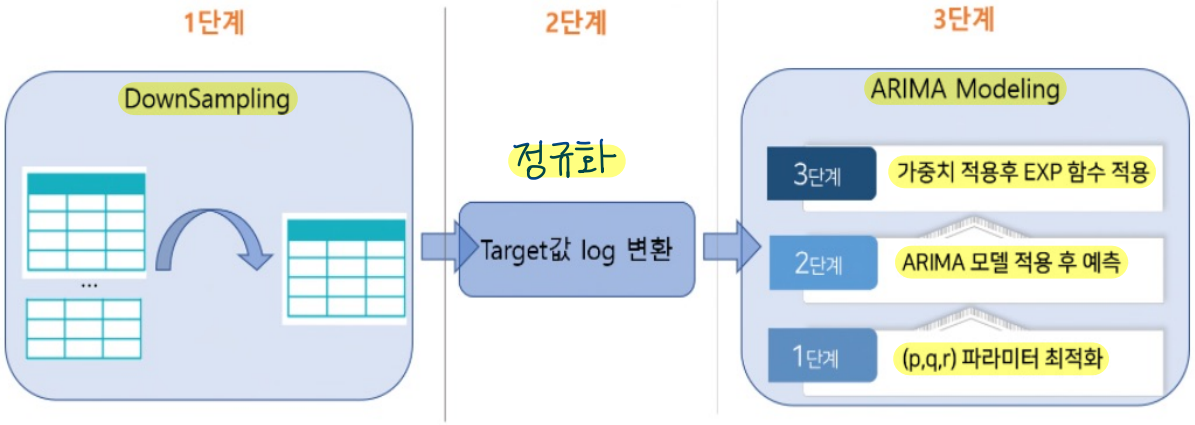
1단계. 각 상점별로 다운샘플링을 진행한다.

2단계. Target 값을 log 변환을 이용해 정규화한다.

3단계. ARIMA 모델링을 진행한다.

In [4]:
```
Image(filename = "C:₩₩Users₩₩kmlam₩₩Documents₩₩auc₩₩그림4.png", width=700, height=700)
```

Out[4]:



앞으로는 용집님의 논리를 단계별로 상세히 설명하고, 또 그것이 어떻게 코드로 구현되었는지를 알아보겠습니다.

Down Sampling

마찬가지로 1회 FUNDA 데이터에서 일별로 되어 있는 매출양을 14일 단위로 다운샘플링한다는 것은 14일별로 묶어서 위 사진과 같은 변환을 해주는 것이다.
데이터를 묶는 개념이기 때문에 다운샘플링을 진행하면, 데이터의 행의 개수가 **감소**한다.

In [6]:
```
Image(filename = "C:₩₩Users₩₩kmlam₩₩Documents₩₩auc₩₩그림2.png", width=500, height=500)
```

Out[6]:

	date	amount
0	2016-08-01	2106
1	2016-08-02	1528
2	2016-08-03	560
3	2016-08-04	1683
4	2016-08-05	1686
5	2016-08-06	2646
6	2016-08-07	645
7	2016-08-08	849
8	2016-08-09	1101
...		
19	2016-08-20	1596
20	2016-08-21	1181
21	2016-08-22	1837
22	2016-08-23	969
23	2016-08-24	1015
24	2016-08-25	1514
25	2016-08-26	1914
26	2016-08-27	512
27	2016-08-28	1019

DownSampling
by 14 Days

	date	amount_sum
0	2016-08-01 ~ 08-14	19311
1	2016-08-15 ~ 08-28	16405

b) 각 상점별로 다운샘플링한 과정

def arima_main에 세팅됨

1) 28일 단위로 다운샘플링을 진행한다.

2) 만약 28일 단위 다운샘플링 이후 행의 개수가 10개를 넘어가면 멈추고 그렇지 않으면 3)번으로 넘어간다.

3) 14일 단위로 다운샘플링을 진행한다.

4) 만약 14일 단위 다운샘플링 이후 행의 개수가 10개를 넘어가면 멈추고 그렇지 않으면 5)번으로 넘어간다.

5) 7일 단위로 다운샘플링을 진행한다.

6) 만약 7일 단위 다운샘플링 이후 행의 개수가 10개를 넘어가면 멈추고 그렇지 않으면 7)번으로 넘어간다.

7) 다운샘플링을 진행하지 않는다.

위의 과정(1번부터 7번)을 각 상점별로 진행한다.

그러면 상점들은 아래의 유형으로 분류될 것이다.

유형 1. 28일 단위로 다운샘플링된 상점

유형 2. 14일 단위로 다운샘플링된 상점

유형 3. 7일 단위로 다운샘플링된 상점

유형 4. 다운샘플링이 되지 않은 상점

c) 다운샘플링을 진행하지 않았을 때와 진행했을 때의 차이

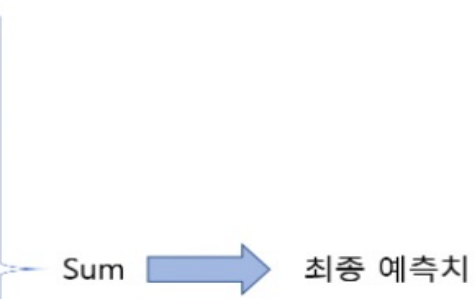
1회 펀다 대회는 미래 100일동안 상점의 매출 합을 예측하는 것이 문제이다.

통상적인 방법은 다운샘플링을 진행하지 않고 아래와 같이 ARIMA 모델을 이용해 각 일별 매출을 예측하고 그렇게 산출된 100개의 매출 예측치들을 모두 더하여 미래 100일동안 상점 매출 총합을 맞추는 것이다.

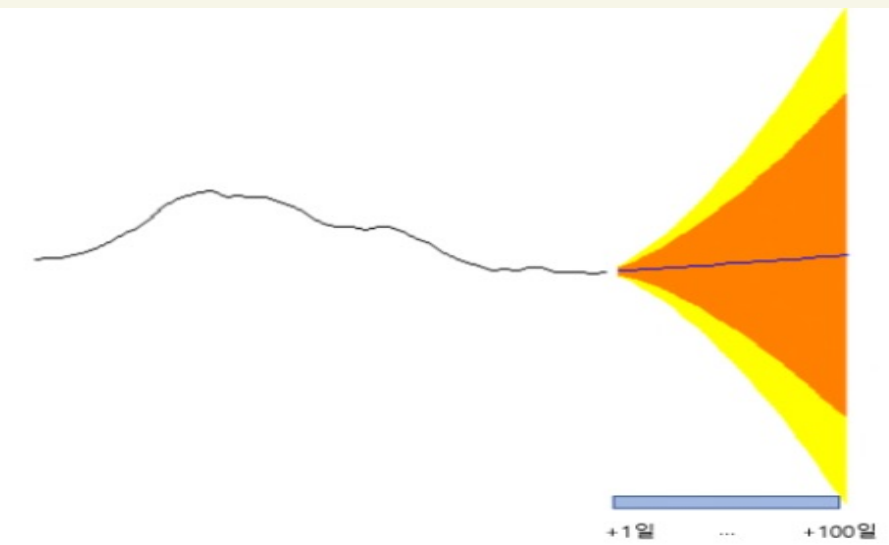
In [7]:
```
Image(filename = "C:₩₩Users₩₩kmlam₩₩Documents₩₩auc₩₩그림5.png", width=400, height=400)
```

Out[7]:

days	amount_pred
+1일	100
+2일	230
+3일	540
+4일	102
+5일	43
+6일	203
+7일	324
...	
+94일	102
+95일	220
+96일	140
+97일	152
+98일	233
+99일	220
+100일	724



그러나 위와 같은 방식으로 예측을 진행하면 생기는 단점이 있다. 바로 예측하려는 **일수가 늘어날 수록** 예측의 **불확실성**이 커진다는 것이다.
아래 그림과 같이 +1일째는 불확실성이 그리 크지 않으나, 일수가 늘어날 수록 예측치의 신뢰구간이 늘어나 불확실성이 점점 커진다.



그러면 증폭하는 불확실성을 줄이려면 어떤 방법이 있을까? 다운샘플링을 활용하면 어느 정도 해결된다. 예를 들어 50일 단위로 과거 데이터를 모두 다운샘플링하고 미래에 대한 예측을 하면 어떻게 될까?

다운샘플링 전에는 +1일부터 +100일까지 각 일별로 매출을 예측했던 반면,
50일 단위로 다운샘플링을 진행하고 예측하면, +1일부터 +50일까지 매출과 +51일부터 +100일까지의 매출을 예측하게 된다.

즉, 예측 대상이 100개였던 것이 다운샘플링 후 **2개**로 줄어든 것이다.

예측대상이 줄어든다는 것은 위의 사진과 같은 문제점을 어느 정도 해결할 수 있는 행위이다.
그러나 다운샘플링을 하는 단위가 너무 크면 시계열에서 중요하게 다루어지는 seasonality가 무시될 수 있기 때문에,
적정한 단위를 고르는 것이 중요하다.

1회 대회 우승자 용집님은, **seasonality를 최대한 보존**하며 예측을 진행하기 위해 일단
28일(4주)을 기준으로 다운샘플링을 진행했으며, 다운샘플링 후 충분한 샘플수가 확보되지 않았을 시에는
14일(2주), 7일(1주)로 단위를 낮추어서 진행했다.

이런 방법은 seasonality를 보장할 뿐만 아니라, 예측 대상이 많음으로써 생기는 불확실성을 어느 정도 해결하는 방법이다.

2단계. Target 값을 log 변환을 이용해 정규화한다.

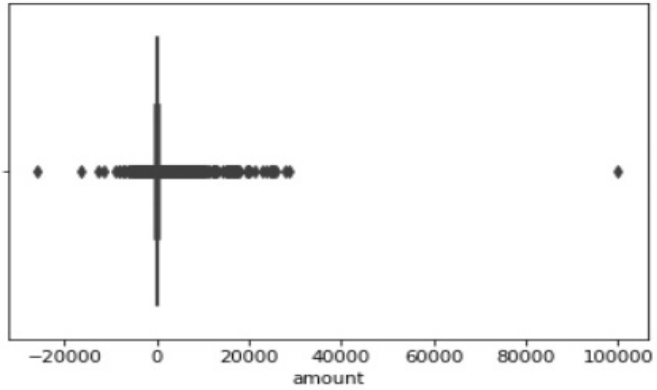
아래 boxplot을 보면 amount(결제 금액)에 **outlier**가 굉장히 많다는 것을 알 수 있다.

용집님은 예측을 함에 있어 **outlier의 영향을 줄이기 위해** 1단계에서 다운샘플링을 진행한 후 각 상점별로 amount를 모두 **log 변환**을 해 주었다.

In [12]:
```
sns.boxplot(test['amount'])
```

Out[12]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c09261a630>
```



3단계. ARIMA 모델링을 진행한다.

a) p,d,q 파라미터 최적화

ARIMA 모델에는 파라미터가 세 가지가 있다. p,d,q이다. 각 파라미터에 대한 설명은 아래와 같다.

p - 자기회귀(Autoregression), 이전 관측치와 현재 관측치와의 관계를 규정

d - 시계열모형을 stationary하게 만들기 위해 차분을 이용하는 것

q - 이전 관측치의 편차와의 관계를 규정

가능한 p,d,q의 경우는 아래와 같다.

1. (1,0,0)
2. (1,0,1)
3. (1,1,0)
4. (1,1,1)
5. (0,0,0)
6. (0,0,1)
7. (0,1,0)
8. (0,1,1)

각각의 parameter 조합에 대하여 ARIMA 모델을 따로 학습시킨다(ARIMA 모델 8개 생성)

8개의 ARIMA 모델 중 AIC 값이 최소인 모델을 선택한다.

**AIC란 아카이케 정보 기준의 준말로, 모델이 기존의 데이터를 얼마나 잘 설명하는지, 그리고 그 모델이 얼마나 간단한지의 두 요소에 대한 평가치이다. 적을수록 최적의 모형이라고 해석할 수 있다.

In [14]:
```
Image(filename = "C:₩₩Users₩₩kmlam₩₩Documents₩₩auc₩₩그림10.png", width=600, height=600)
```

Out[14]: