

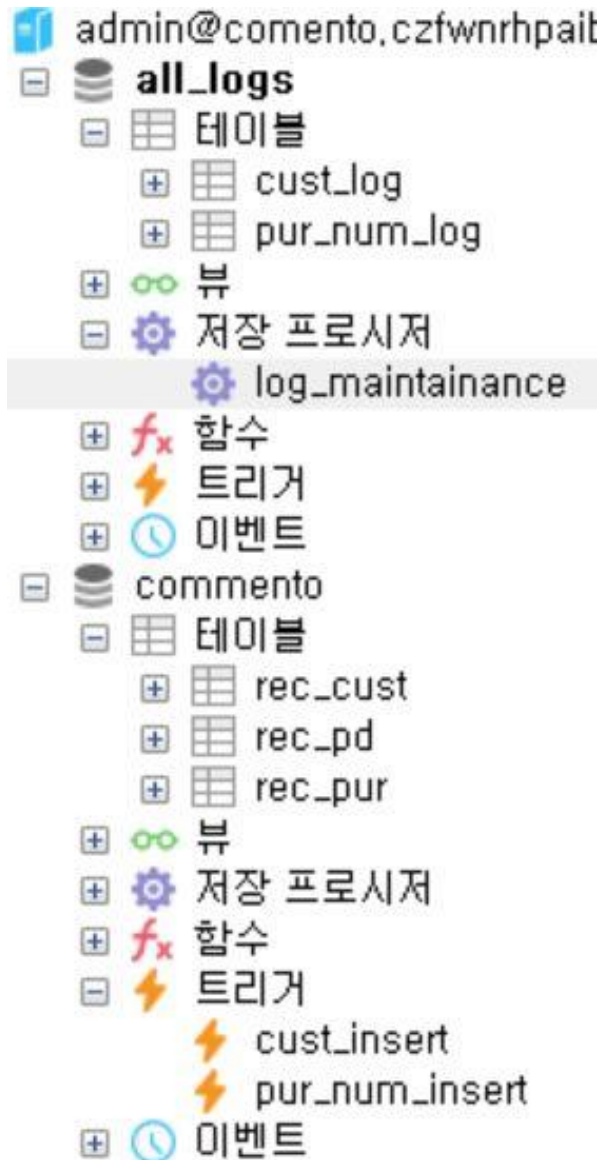
과제 3

이상데이터 검출 및 품질보증 시스템 생성

목차

1. DB 설명
2. Log DB의 table 설명 및 이상 데이터 설명
3. (수동)이상데이터에 대해 직접 all_logs DB내 log table에 insert
4. (자동)이상데이터에 대해 Trigger 사용
5. 점검 프로시저 작성

1. DB 설명



all_logs : 이상 데이터 로그 모음 DB

comento : 고객추천을 위한 insert 데이터 모음 DB

2. 이상 데이터 설명 및 관련 log table

1. 상품을 2회 산 고객

전체 상품 중 총 상품을 산 횟수가 2회인 고객을 이상 데이터로 간주

Log table : pur_num_log

2. rec_cust에 insert당시 이상 데이터 로그

- 1) 휴대폰 번호 이상 : 14자리가 아니거나 숫자가 아닌경우
- 2) Gender이상 : 1혹은2가 아닌 숫자인 경우
- 3) Birth이상 : 8자리 숫자가 아니거나 실존하는 날짜가 아닌경우

Log table : cust_log

2. 이상 데이터 설명 및 관련 log table

이상 데이터 1에 대한 로그 테이블 생성 및 확인

* 이상 데이터 1 : 상품을 2회 산 고객, pur_num_log

```
USE all_logs;
```

```
--log table : 고객이 상품을 산 횟수 2회 이상이면 이 로그 테이블에 남김
```

```
CREATE TABLE `pur_num_log`
```

```
(  
  `cust_id` VARCHAR(30) NOT NULL
```

```
COMMENT '고객아이디',
```

```
  `data_input_time` DATE NOT NULL
```

```
COMMENT '데이터입력일시',
```

```
  `pur_num` VARCHAR(30) NOT NULL
```

```
COMMENT '상품구매횟수',
```

```
  `data_input_nm` VARCHAR(30) NOT NULL
```

```
COMMENT '데이터입력자'
```

```
)`all_logs`
```

```
COMMENT = '로그테이블';
```

```
ALTER TABLE `pur_num_log`
```

```
ADD CONSTRAINT `엔터티2_PK` PRIMARY KEY (  
  `cust_id`);
```

```
1 USE all_logs;
```

```
2
```

```
3 DESC pur_num_log;
```

1 결과 2 프로파일러 3 메시지 4 테이블 데이터 5 정보							
(읽기 전용)							
<input type="checkbox"/>	Field	Type		Null	Key	Default	Extra
<input type="checkbox"/>	cust_id	var...	11B	NO	PRI	(NULL)	OK
<input type="checkbox"/>	data_input_time	date	4B	NO		(NULL)	OK
<input type="checkbox"/>	pur_num	var...	11B	NO		(NULL)	OK
<input type="checkbox"/>	data_input_nm	var...	11B	NO		(NULL)	OK

2. 이상 데이터 설명 및 관련 log table

이상 데이터 2에 대한 로그 테이블 생성 및 확인

* 이상 데이터 2 : rec_cust insert당시 이상 데이터 로그, cust_log

```
--log table2 : 고객정보insert문제
CREATE TABLE `cust_log`
(
  `cust_id` VARCHAR(30) NOT NULL
  COMMENT '고객아이디',
  `log_col` varchar(30) not NULL
  comment '이상column',
  `log_content` VARCHAR(30)
  COMMENT '이상데이터내용',
  `data_input_time` DATE NOT NULL
  COMMENT '데이터입력일시',
  `data_input_nm` VARCHAR(30) NOT NULL
  COMMENT '데이터입력자',
  `data_mod_time` DATE NOT NULL
  COMMENT '데이터수정일시',
  `data_mod_nm` VARCHAR(30) NOT NULL
  COMMENT '데이터수정자'
)
COMMENT = '고객로그테이블';
```

```
ALTER TABLE `cust_log`
ADD CONSTRAINT `엔터티2_PK1` PRIMARY KEY (
`cust_id`,`log_col`);
```

```
1 USE all_logs;
2
3 DESC cust_log;
```

1 결과

2 프로파일러

3 메시지

4 테이블 데이터

5 정보

</

이상 데이터 column 이름과 그 내용 저장

* 가능 column : gender, birth, cell_no

3. (수동) 직접 all_logs DB내 log table에 insert

이상 데이터 1 : 이상 고객 확인 후, 직접 table에 insert

상품을 산 횟수가 2회인 고객 확인

```
1  USE commento;
2
3  SELECT cust_id, COUNT(1)
4  FROM `rec_pur`
5  GROUP BY cust_id
6  HAVING COUNT(1) > 1
7  ;
8
9
```

1 결과	2 프로파일러	3 메시지
(읽기 전용)		
<input type="checkbox"/> cust_id	COUNT(1)	
<input type="checkbox"/> ID03	2	

Table에 insert 후 확인

```
#이상데이터로그 : 상품 구매 횟수 2회 이상인 고객
INSERT INTO
all_logs.pur_num_log(`cust_id`,`data_input_time`,`pur_num`,
`data_input_nm`)
SELECT cust_id, data_input_time,COUNT(1),data_input_nm
FROM `rec_pur`
GROUP BY cust_id
HAVING COUNT(1) > 1
;
```

```
1  USE all_logs;
2
3  SELECT * FROM pur_num_log;
```

1 결과	2 프로파일러	3 메시지	4 테이블 데이터	5 정보
(읽기 전용)				
<input type="checkbox"/> cust_id	data_input_time	pur_num	data_input_nm	
<input type="checkbox"/> ID03	2020-10-30	2	system	


3. (수동) 직접 all_logs DB내 log table에 insert

이상 데이터 2 : 이상 고객 확인 후, 직접 table에 insert

이상 데이터 확인 : 핸드폰 번호 이상한 경우

Table에 insert 후 확인

```
1  USE commento;
2
3  SELECT cust_id, cust_nm, cell_no
4  FROM `rec_cust`
5  WHERE cell_no NOT REGEXP '[0-9]{2,3}-[0-9]{3,4}-[0-9]{4}' OR LENGTH(`cell_no`) > 14
6  ;
7
```

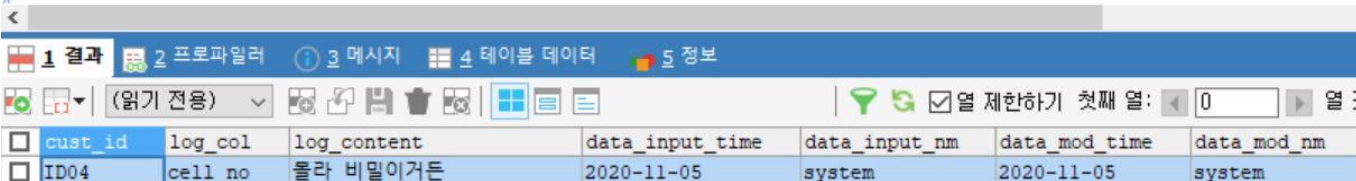


The screenshot shows a database interface with a query result table. The table has three columns: cust_id, cust_nm, and cell_no. The first row of data shows cust_id 'ID04', cust_nm 'OPPS', and cell_no '올라 비밀이거든'.

cust_id	cust_nm	cell_no
ID04	OPPS	올라 비밀이거든

```
INSERT INTO all_logs.cust_log(`cust_id`,`log_col`,`log_content`,
`data_input_time`,`data_input_nm`,`data_mod_time`,`data_mod_nm`)
SELECT cust_id, 'cell_no',
cell_no,data_input_time,data_input_nm,data_mod_time,data_mod_nm
FROM `rec_cust`
WHERE cell_no NOT REGEXP '[0-9]{2,3}-[0-9]{3,4}-[0-9]{4}' OR
LENGTH(`cell_no`) > 14
;
```

```
1  USE all_logs;
2
3  SELECT * FROM cust_log;
```



The screenshot shows a database interface with a query result table. The table has seven columns: cust_id, log_col, log_content, data_input_time, data_input_nm, data_mod_time, and data_mod_nm. The first row of data shows cust_id 'ID04', log_col 'cell_no', log_content '올라 비밀이거든', data_input_time '2020-11-05', data_input_nm 'system', data_mod_time '2020-11-05', and data_mod_nm 'system'.

cust_id	log_col	log_content	data_input_time	data_input_nm	data_mod_time	data_mod_nm
ID04	cell_no	올라 비밀이거든	2020-11-05	system	2020-11-05	system

4. (자동) trigger 사용

사용 위한 set up : DB 인스턴스 파라미터 일부 변경

log_bin_trust_function_creators의 값을 0에서 1로 변경

```
1  SHOW GLOBAL VARIABLES LIKE 'log_bin_trust_function_creators';
2
```



Variable_name	Value
log_bin_trust_function_creators	ON

과정 : AWS DB instance의 구성-파라미터 그룹-파라미터 새로 생성-log_bin_trust_function_creators값 1로 변경
- 데이터베이스 화면으로 와서 해당 instance 수정 후 재부팅

4. (자동) trigger 사용

이상데이터 1 : trigger생성

```
DELIMITER $$

CREATE
  TRIGGER `commento`.`pur_num_insert` AFTER INSERT
  ON `commento`.`rec_pur`
  FOR EACH ROW BEGIN
  DECLARE pur_num_ INT;

  SELECT COUNT(1) INTO pur_num_
  FROM `rec_pur`
  GROUP BY cust_id
  HAVING cust_id = new.cust_id;

  IF pur_num_ > 1 THEN
    INSERT INTO
all_logs.pur_num_log(`cust_id`,`data_input_time`,`pur_num`,
`data_input_nm`)
VALUES
(new.cust_id,new.data_input_time,pur_num_,new.data_input_nm)
ON DUPLICATE KEY UPDATE
  cust_id=new.cust_id,
  data_input_time=new.data_input_time,
  pur_num=pur_num_,
  data_input_nm=new.data_input_nm
  ;
  END IF;

END$$

DELIMITER ;
```

- Rec_pur에 대한 insert발생시 trigger작동
- Per_num_변수 : 입력들어온 데이터가 전체 상품 구매한 횟수
- Pur_num_이 2이상인 경우, 이상 데이터이므로, pur_num_log 테이블에 insert 혹은 update하여 로그 기록한다.

4. (자동) trigger 사용

이상데이터 1 : 추가 데이터 삽입 및 정상작동 확인

데이터 삽입 :

ID01,ID03은 본래는 상품을 각자 1,2개씩 구매하였으나,
데이터 삽입으로 상품을 2,3개씩 구매하게 된 상황

```
INSERT INTO `rec_pur`(`cust_id`, `pd_id`, `data_input_time`,  
`data_input_nm`)  
VALUES ('ID01', 'PD02', NOW(), 'system'),  
('ID03', 'PD02', NOW(), 'system')  
;
```

정상 작동 확인

```
1 USE all_logs;  
2  
3 SELECT * FROM pur_num_log;
```

1 결과 2 프로파일러 3 메시지 4 테이블 데이터 5 정보				
(읽기 전용)				
<input type="checkbox"/>	cust_id	data_input_time	pur_num	data_input_nm
<input type="checkbox"/>	ID01	2020-11-05	2	system
<input type="checkbox"/>	ID03	2020-11-05	3	system

4. (자동) trigger 사용

이상데이터 2 : trigger생성

- Cust_rec에 대한 insert발생시 trigger작동
- Cell_issue변수 : cell_no가 이상이 있는지
(이상 : 1이상, 정상 : 0)
- Birth_issue변수 : birth가 이상이 있는지
- Gender_issue변수 : gender가 이상이 있는지
- 각각의 변수에 대해서 0이상이면 이상으로 간주하고 insert혹은 update하여 로그 기록
- cust_log의 private key는 cust_id, cust_issue로 3개의 issue가 모두 있다면 각각 모두에 대해 log남김

```
DELIMITER $$

CREATE
  TRIGGER 'commento'. 'cust_insert' AFTER INSERT
  ON 'commento'. 'rec_cust'
  FOR EACH ROW BEGIN
    DECLARE cell_issue INT;
    DECLARE birth_issue INT;
    DECLARE gender_issue INT;

    SET cell_issue = 0;
    SET birth_issue = 0;
    SET gender_issue = 0;

    SELECT COUNT(1) INTO cell_issue
    FROM 'rec_cust'
    WHERE new.cell_no NOT REGEXP '[0-9]{2,3}-[0-9]{3,4}-[0-9]{4}' OR LENGTH(new.cell_no) > 14
    ;

    SELECT COUNT(1) INTO gender_issue
    FROM 'rec_cust'
    WHERE new.gender!=1 AND new.gender!=2
    ;

    SELECT COUNT(1) INTO birth_issue
    FROM 'rec_cust'
    WHERE LENGTH(new.birth) != 8 OR new.birth NOT REGEXP '^(19|20)[0-9][0-9]([01-9]|1[012])([01-9]|1[12])[0-9][3[0-1])$'
    ;

    IF cell_issue > 0 THEN
      INSERT INTO all_logs.cust_log('cust_id','log_col','log_content',
        'data_input_time','data_input_nm','data_mod_time','data_mod_nm')
      VALUES (new.cust_id, 'cell_no', new.cell_no,new.data_input_time,new.data_input_nm,new.data_mod_time,new.data_mod_nm)
      ON DUPLICATE KEY UPDATE
        cust_id = new.cust_id,
        log_col = 'cell_no',
        log_content = new.cell_no,
        data_input_time = new.data_input_time,
        data_input_nm = new.data_input_nm,
        data_mod_time = new.data_mod_time,
        data_mod_nm = new.data_mod_nm;
    END IF;

    IF gender_issue > 0 THEN
      INSERT INTO all_logs.cust_log('cust_id','log_col','log_content',
        'data_input_time','data_input_nm','data_mod_time','data_mod_nm')
      VALUES (new.cust_id, 'gender', new.gender,new.data_input_time,new.data_input_nm,new.data_mod_time,new.data_mod_nm)
      ON DUPLICATE KEY UPDATE
        cust_id = new.cust_id,
        log_col = 'gender',
        log_content = new.gender,
        data_input_time = new.data_input_time,
        data_input_nm = new.data_input_nm,
        data_mod_time = new.data_mod_time,
        data_mod_nm = new.data_mod_nm;
    END IF;

    IF birth_issue > 0 THEN
      INSERT INTO all_logs.cust_log('cust_id','log_col','log_content',
        'data_input_time','data_input_nm','data_mod_time','data_mod_nm')
      VALUES (new.cust_id, 'birth', new.birth,new.data_input_time,new.data_input_nm,new.data_mod_time,new.data_mod_nm)
      ON DUPLICATE KEY UPDATE
        cust_id = new.cust_id,
        log_col = 'birth',
        log_content = new.birth,
        data_input_time = new.data_input_time,
        data_input_nm = new.data_input_nm,
        data_mod_time = new.data_mod_time,
        data_mod_nm = new.data_mod_nm;
    END IF;

  END$$

DELIMITER ;
```

4. (자동) trigger 사용

이상데이터 2 : 추가 데이터 삽입 및 정상작동 확인

데|이|터 삽입 :

ID05 – gender issue 발생해야함

ID06 – cell_no, birth issue 발생해야함

```
INSERT INTO `rec_cust`(`cust_id`, `cust_nm`, `gender`, `birth`, `cell_no`, `ap_path_cd`, `data_input_time`, `data_input_nm`,
`data_mod_time`, `data_mod_nm`)
VALUES ('ID05', 'YUJIN', '3', '19800104', '010-2222-1111', 'P02', NOW(), 'system', NOW(), 'system'),
('ID06', 'GANG', '1', '000920', '010-3212', 'P01', NOW(), 'system', NOW(), 'system')
;
```

정상 작동 확인

```
1  USE all_logs;
2
3  SELECT * FROM cust log;
```

<input type="checkbox"/> cust_id	log_col	log_content	data_input_time	data_input_nm	data_mod_time	data_mod_nm
<input type="checkbox"/> ID04	cell_no	물라 비밀이거든	2020-11-05	system	2020-11-05	system
<input type="checkbox"/> ID05	gender	3	2020-11-05	system	2020-11-05	system
<input type="checkbox"/> ID06	birth	000920	2020-11-05	system	2020-11-05	system
<input type="checkbox"/> ID06	cell_no	010-3212	2020-11-05	system	2020-11-05	system

5. 점검 프로시저 작성

All_logs의 두 log table에 대하여 결과 table을 보여주는 프로시저 작성

```
DELIMITER $$  
  
USE `all_logs`$$  
  
DROP PROCEDURE IF EXISTS `log_maintenance`$$  
  
CREATE DEFINER=`admin`@`%` PROCEDURE `log_maintenance`()  
BEGIN  
    SELECT * FROM cust_log;  
    SELECT * FROM pur_num_log;  
  
    SELECT * FROM commento.rec_cust  
    WHERE rec_cust.cust_id = ANY(SELECT cust_id FROM cust_log);  
  
    SELECT * FROM commento.rec_pur  
    WHERE rec_pur.cust_id = ANY(SELECT cust_id FROM pur_num_log);  
  
END$$  
  
DELIMITER ;
```

Log_table의 table들을 보여주고,
commento DB내의 rec_cust, rec_pur에서 이
상 데이터를 골라내어 보여주는 procedure

5. 점검 프로시저 작성

프로시저를 호출하였을 때 결과

1 CALL log_maintenance;

1 결과 2 프로파일러 3 결과 4 메시지 5 테이블 데이터 6 정보

(읽기 전용)

열 제한하기 첫째 열: 0 열 갯

<input type="checkbox"/> cust_id	log_col	log_content	data_input_time	data_input_nm	data_mod_time	data_mod_nm
<input type="checkbox"/> ID04	cell_no	몰라 비밀이거든	2020-11-05	system	2020-11-05	system
<input type="checkbox"/> ID05	gender	3	2020-11-05	system	2020-11-05	system
<input type="checkbox"/> ID06	birth	000920	2020-11-05	system	2020-11-05	system
<input type="checkbox"/> ID06	cell_no	010-3212	2020-11-05	system	2020-11-05	system

<input type="checkbox"/>	cust_id	data_input_time	pur_num	data_input_nm
<input type="checkbox"/>	ID01	2020-11-05	2	system
<input type="checkbox"/>	ID03	2020-11-05	3	system

All_logs의 두 테이블의 내용

5. 점검 프로시저 작성

프로시저를 호출하였을 때 결과

```
1 CALL log_maintenance;
```

1 결과

2 프로파일러

3 결과

4 결과

5 결과

6 메시지

7 테이블 데이터

8 정보

(읽기 전용)

</

```
1 CALL log_maintenance;
```

1 결과	2 프로파일러	3 결과	4 결과	5 결과	6 메시지	7 테이블 데이터	8 정보
<div>(읽기 전용)</div>							
<input type="checkbox"/>	cust_id	pd_id	data_input_time	data_input_nm			
<input type="checkbox"/>	ID01	PD01	2020-10-30	system			
<input type="checkbox"/>	ID01	PD02	2020-11-05	system			
<input type="checkbox"/>	ID03	PD01	2020-10-30	system			
<input type="checkbox"/>	ID03	PD02	2020-11-05	system			
<input type="checkbox"/>	ID03	PD03	2020-10-30	system			

Log내의 이상 데이터를 commento내의 테이블에서 확인 : 이상 데이터의 전체 확인 가능

기대효과

1. Trigger를 사용한다면, insert할 때 자동적으로 log를 남겨주므로, 따로 점검 시간을 만든 후 log에 필요한 데이터를 insert하기위한 인력 및 시간소모를 줄여준다.
2. 점검 프로시저를 사용한다면, sql문을 따로따로 입력할 필요 없이 프로시저만 호출하면 되므로, 그에 따른 시간소모도 줄여준다.

기대효과 : 점검마다 코드량 비교

Trigger, procedure 사용 전 코드
: 각각에 대하여 실행해야 함

```
--상품들 2개 선택 고객 검색
SELECT cust_id, COUNT(1)
FROM `rec_pur`
GROUP BY cust_id
HAVING COUNT(1) > 1
;

--이상 고객 log에 넣기
INSERT INTO all_logs.pur_num_log('cust_id','data_input_time','pur_num',
'data_input_nm')
SELECT cust_id, data_input_time,COUNT(1),data_input_nm
FROM `rec_pur`
GROUP BY cust_id
HAVING COUNT(1) > 1
;

--이상데이터 검증1 : cell_num이 이상할 때
SELECT cust_id, cust_nm, cell_no
FROM `rec_cust`
WHERE cell_no NOT REGEXP '[0-9]{2,3}-[0-9]{3,4}-[0-9]{4}' OR
LENGTH('cell_no') > 14
;

--이상데이터 검증2 : gender이상할 때
SELECT cust_id, cust_nm, gender
FROM `rec_cust`
WHERE gender!=1 AND gender!=2;

--이상데이터 검증3 : birth이상할 때
SELECT cust_id, cust_nm, birth
FROM `rec_cust`
WHERE LENGTH(birth) != 8 or birth NOT REGEXP '^(19|20)[0-9][0-9]([01-9]|1[012])([01-9]|[12][0-9]|3[0-1])$'
;

--이상 고객 log에 넣기
INSERT INTO all_logs.cust_log('cust_id','log_col','log_content',
'data_input_time','data_input_nm','data_mod_time','data_mod_nm')
SELECT cust_id, 'cell_no',
cell_no,data_input_time,data_input_nm,data_mod_time,data_mod_nm
FROM `rec_cust`
WHERE cell_no NOT REGEXP '[0-9]{2,3}-[0-9]{3,4}-[0-9]{4}' OR
LENGTH('cell_no') > 14
;

#점검
SELECT * FROM cust_log;
SELECT * FROM pur_num_log;

SELECT * FROM commento.rec_cust
WHERE rec_cust.cust_id = ANY(SELECT cust_id FROM cust_log);

SELECT * FROM commento.rec_pur
WHERE rec_pur.cust_id = ANY(SELECT cust_id FROM pur_num_log);
```

Trigger, procedure 사용 후 코드
: 점검시 한문장만 사용하면 됨

```
CALL log_maintenance;
```

추가적으로 진행할 수 있는 것

1. Trigger에서 데이터 삭제가 있다면 삭제한 데이터에 대해서도 trigger생성
2. 점검 프로시저 구체화 : 트리거로 인한 log변화 말고도 log 변화 있는지 확인하는 추가 코드 필요할 것