유치원 유아 모집 등록 관리 시스템 최종 보고서

정보 시스템 설계 2022 1R 팀프로젝트

2019131438 송은서
2019170839 박지영
2019170853 장예린

목차 A table of contents

1 시스템 정의 및 개발목적

시스템 정의

유치원 유아 모집 등록 관리 시스템

시스템 개발 목적

1. 아이의 유치원 지원을 위해 학부모들에게 편리한 정보 제공
2. 유치원 원아 선발 과정에 있어 효율성 증대
3. 유치원 정원 미달, 과열 등의 정보 제공을 통해 지원 과정 중 원아 몰림 현상 발생 방지

1 시스템 정의 및 개발목적

시스템 가정

① 학부모 1명 당 유아 1명

② 선발 전/ 선발 후 ~ 등록기간 전/ 등록기간 후 ~ 추가모집의 과정

③ 학부모의 희망 조건에 따른 적합 유치원은, 조건을 만족하는 유치원 목록 중 3개가 시스템 상에서 랜덤으로 선택

④ 대기 포기는 오직 타 유치원 등록에 의해 자동으로 이루어짐

⑤ 학부모는 관심이 있는 유치원에 대해 '관심' 표시를 할 수 있다.

⑥ 모집 일정 가정은 다음과 같음

희망 유치원 지원 기간	2023-05-01 00:00:00 ~ 2023-05-08 00:00:00
선발 결과 발표 및 등록/포기 기간	2023-05-08 00:00:01~ 2023-05-18 00:00:00
추가모집 기간	2023-05-18 00:00:01~ 2023-05-21 00:00:00
추가모집 결과 발표	2023-05-21 00:00:01

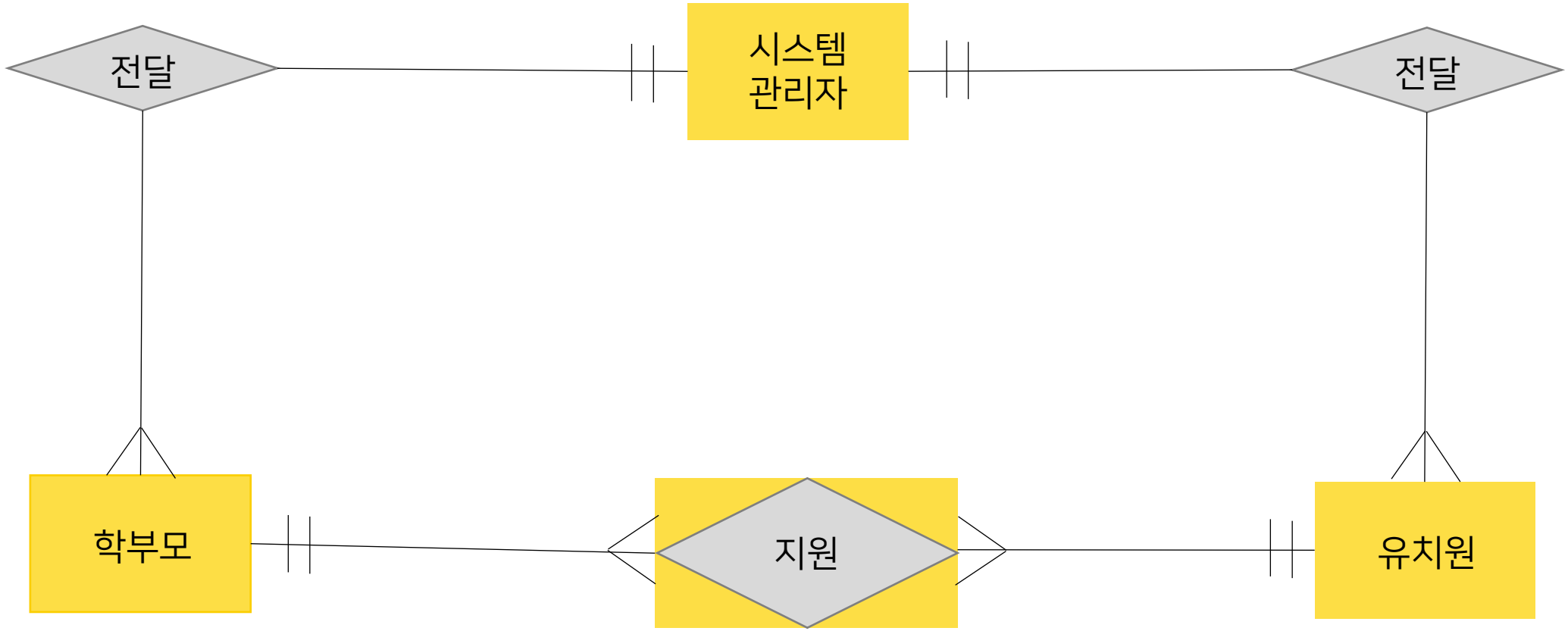
⑦ 선발 결과 발표와 동시에 등록/포기 기간이 시작되며, 등록/포기기간 종료와 동시에 추가모집기간이 시작됨

2 사용자 요구분석

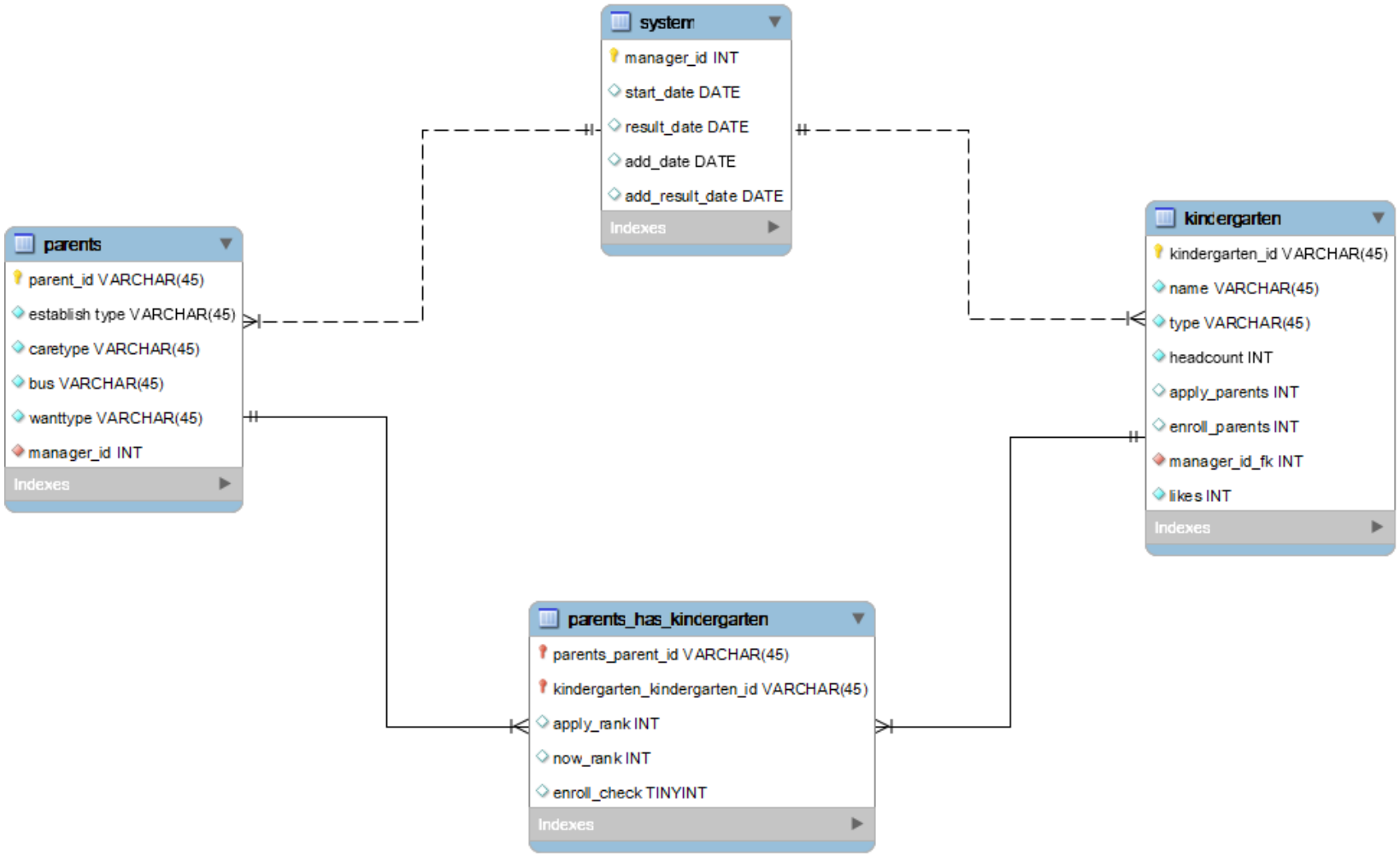
사용자 요구분석

	요구 사항	Query
학부모	▪ (선발 전) 원하는 조건에 따른 유치원 목록을 알고 싶다. ▪ (선발 전) 지원하고자 하는 유치원의 경쟁률을 알고 싶다. ▪ (선발 후~등록기간 종료 전) '대기상태'인 유치원에서 나의 현재 대기 순위를 알고 싶다. ▪ (선발 후 ~ 등록기간 종료 전) 관심 유치원 목록 내에서 결원이 발생했는지 알고 싶다.	▪ 학부모 A의 선호 조건에 적합하는 유치원 목록은 무엇인가? ▪ 유치원 B의 정원 대비 '관심 수'의 비율은 얼마인가? ▪ 내가 '대기 상태'인 유치원에서의 내 대기 순위는 얼마인가? ▪ 유치원 B에서 결원이 발생했는가?
유치원	▪ (선발 후 ~ 등록기간 전) 현재 대기 중인 학부모의 수를 알고 싶다. ▪ (등록기간 후 ~ 추가모집 마감) 해당 모집의 미달 여부를 알고 싶다.	▪ 유치원 B에 대기 중인 학부모의 수는? ▪ 유치원 B의 정원보다 등록인원이 적은가?
시스템 관리자	▪ 유치원 모집 일정에 대해 안내하고 싶다.	▪ 현재는 유치원 모집 일정의 어느 단계인가?

3 ERD



3 ERD in MySQL



4 Relation Schema

5 Logic Modeling : Query 1

(USER : 학부모, 기간 : 선발 전)

Q1. 학부모 A의 선호 조건에 적합하는 유치원 목록은 무엇인가?

Decision Table >>

Conditions	rules															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
설립유형	국공립	국공립	국공립	국공립	사립	사립	사립	사립	국공립	국공립	국공립	국공립	사립	사립	사립	사립
돌봄유형	기본운영	온종일	저녁	아침	기본운영	온종일	저녁	아침	기본운영	온종일	저녁	아침	기본운영	온종일	저녁	아침
통학차량	O	O	O	O	O	O	O	O	X	X	X	X	X	X	X	X
Type	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P

Pseudo Code >>

```
select 유치원.유치원이름
from 유치원
where 유치원.유치원TYPE = type
```

5 Logic Modeling : Query 2

(USER : 학부모, 기간 : 선발 전)

Q2. 유치원 B의 경쟁률 및 선호율은 어떻게 되는가?

Pseudo Code >>

X = 입력받은 지원하려는 유치원 이름
FIND X IN 유치원
PRINT X.관심수/X.정원
PRINT X.관심수/NROW(학부모)

5 Logic Modeling : Query3

(USER : 학부모, 기간 : 선발 후~등록기간 종료 전)

Q3. 내가 '대기 상태'인 유치원에서의 내 대기 순위는 얼마인가?

Pseudo Code >>

```
SELECT 유치원.유치원이름,

FROM 지원, 유치원
WHERE 지원.선발결과(최초대기순위) > 0
        AND 지원.등록여부 = Null
        AND 유치원.유치원ID = 지원.유치원ID
        AND 지원.학부모ID = userID ;
```

5 Logic Modeling : Query 4

(USER : 학부모, 기간 : 선발 후~등록기간 종료 전)

Q4. 유치원 B에서 결원이 발생했는가?

Pseudo Code >>

<등록 발생 시, update>

```
IF 지원.유치원ID == X.유치원ID:
   유치원.등록학부모수 = SUM(지원.등록여부)
```

<조회 및 return>

```
if X.정원 수 - 유치원.등록학부모수 > 0 :
        print 'X 유치원에서 (X.정원수 - X.등록학부모수)명의 결원 발생'
else:
        print('결원이 발생하지 않음')
```

5 Logic Modeling : Query 5

(USER : 유치원, 기간 : 선발 후~등록기간 종료 전)

Q5. 유치원 B에 대기 중인 학부모의 수는?

Pseudo Code >>

```
X = 입력된 유치원이름
FIND X IN 유치원
순위 = MAX(X.현재대기순위)
if 순위 > 0 :
PRINT ' {순위} 명이 대기중입니다.'
else:
        PRINT ' 대기인원이 없습니다.'
```

5 Logic Modeling : Query 6

(USER : 유치원, 기간 : 등록기간 후 ~ 추가모집 마감)

Q6. 유치원 B의 정원보다 등록인원이 적은가?

Pseudo Code >>

```
X = 입력된 유치원 이름
FIND X IN 유치원
IF X.정원 > X.등록학부모수 :
    PRINT '등록 기간동안 미달'
ELSE :
    PRINT '등록 기간동안 정원을 충족함'
```

5 Logic Modeling : Query 7

(USER : 시스템 관리자, 기간 : 항상)

Q7. 현재는 유치원 모집 일정의 어느 단계인가?

Pseudo Code >> SELECT NOW()

```
IF NOW() IN RANGE(시스템관리자.선발시작날짜, 시스템관리자.선발결과발표날짜) :
    PRINT '유치원 지원 기간입니다.'
    PRINT '선발결과는 (시스템관리자.선발결과발표날짜)에 발표됩니다.'
ELIF NOW() IN RANGE(시스템관리자.선발결과발표날짜,추가모집시작날짜) :
    PRINT '선발 결과가 발표되었습니다. 등록/포기 기간입니다.'
    PRINT '등록 기간은 (시스템관리자.추가모집시작날짜)까지입니다.'
ELIF NOW() IN RANGE(시스템관리자.추가모집시작날짜) :
    PRINT '등록/포기 기간이 지났습니다. 추가 모집 기간입니다.'
ELSE :
    PRINT '유치원 모집 전형 기간이 아닙니다.'
```

6 Database

학부모 Table (parents)

```
1 • SELECT * FROM mydb.parents;
```

parent_id	establish type	caretype	bus	wanttype	manager_id
kid001	public	standard	o	A	1
kid002	public	all	o	B	1
kid003	public	night	o	C	1
kid004	public	morning	o	D	1
kid005	public	standard	o	A	1
kid006	public	all	o	B	1
kid007	public	night	o	C	1
kid008	public	morning	o	D	1
kid009	public	standard	o	A	1
kid010	public	all	o	B	1
kid011	public	night	o	C	1

[학부모ID] [설립유형] [돌봄유형] [버스유무] [선호유치원 TYPE] [recordID]

유치원 Table (kindergarten)

```
1 • SELECT * FROM mydb.kindergarten;
```

kindergarten_id	name	type	headcount
kind001	love	A	9
kind002	meme	B	7
kind003	pen	C	3

[유치원ID] [유치원이름] [유치원TYPE] [정원수]

apply_parents	enroll_parents	manager_id_fk	likes
0	0	1	40
0	0	1	30
0	0	1	25

[지원학부모수] [등록학부모수] [recordID] [관심수]

6 Database

지원 Table (parents_has_kindergarten)

```
1 • SELECT * FROM mydb.parents_has_kindergarten;
```

parents_parent_id	kindergarten_kindergarten_id
kid001	kind001
kid002	kind002
kid003	kind003

[학부모ID] [유치원ID]

apply_rank	now_rank	enroll_check
0	0	1
0	0	1
0	0	1

[최초대기순위] [현재대기순위] [등록여부]

시스템 관리자 Table(system)

```
1 • SELECT * FROM mydb.system;
```

manager_id	start_date	result_date
1	2023-05-01	2023-05-08

[recordID] [선발시작날짜] [선발결과발표날짜]

add_date	add_result_date
2023-05-18	2023-05-21

[추가모집시작날짜] [추가모집결과날짜]

7 시스템 운영 Procedure

'Applykinder' Procedure (지원)

: 학부모가 유치원에 지원할 경우, 해당 지원 정보를 지원 table(parents_has_kindergarten)에 insert 하는 지원 기능을 수행한다.

① SQL CODE

```
1   #지원 프로시저
2   drop procedure if exists Applykinder;
3   delimiter //
4   create procedure Applykinder(
5   IN parentsid varchar(45),
6   IN kinderid varchar(45))
7
8   #지원이 작성된 학부모의 지원횟수가 3회를 넘는다면 지원 불가.
9   IF (select count(pk.kindergarten_kindergarten_id) from parents_has_kindergarten pk
10      where pk.parents_parent_id = parentsid) = 3
11  then begin
12      select 'already applied 3 kindergarten';
13  end;
14
15  ELSE begin
16      insert into parents_has_kindergarten values (parentsid, kinderid, NULL, NULL, NULL);
17  end;
18  end if; //
19  delimiter ;
```

② 실행 결과

```
21  call Applykinder('kid99', '혜화유치원');
22  call Applykinder('kid99', '병설유치원');
23  call Applykinder('kid99', '호그와트유치원');
24  call Applykinder('kid99', '고려유치원');
```

kid047	kind011	1	1	0
kid99	병설유치원	NULL	NULL	NULL
kid99	혜화유치원	NULL	NULL	NULL
kid99	호그와트유치원	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL

3건의 지원 정보가 지원 Table에 입력된 결과가 확인된다.

already applied 3 kindergarten
already applied 3 kindergarten

3건의 지원을 모두 완료했다면,
지원을 마쳤다는 string을 보여준다.
이 경우, 값이 입력되지 않는다.

7 시스템 운영 Procedure

'Enrollkinder' Procedure (등록 및 대기순위 업데이트)

: 등록이 발생하면 등록 가능 여부를 확인하고 상태 출력 및 등록 시, 대기순위를 자동으로 업데이트하는 기능을 수행한다.

① SQL CODE

```
1   #Enroll&Update Procedure : 등록이 발생하면 등록가능 여부를 확인하고 상태출력 및 등록 시, 대기순위 업데이트
2   drop procedure if exists Enrollkinder;
3   delimiter //
4   create procedure Enrollkinder(IN parentsid varchar(45),
5                                 IN kinderid varchar(45))
6
7   begin
8       declare wait_rank int;
9       SELECT pk.now_rank INTO wait_rank FROM parents_has_kindergarten pk
10      WHERE pk.parents_parent_id = parentsid and pk.kindergarten_kindergarten_id = kinderid;
11
12      drop temporary table if exists temp_forgive;
13      create temporary table temp_forgive( parentsid varchar(45), forgive varchar(45));
14      Insert into temp_forgive(parentsid, forgive)
15      select pk.parents_parent_id, pk.kindergarten_kindergarten_id
16      from parents_has_kindergarten pk
17      where pk.parents_parent_id = parentsid and pk.kindergarten_kindergarten_id != kinderid;
```

Enrollkinder procedure 정의

등록하지 않을 유치원 정보를 저장하는 temporary table temp_forgive를 생성. 이를 이용해 대기순위 update를 구현함

7 시스템 운영 Procedure

'Enrollkinder' Procedure (등록 및 대기순위 업데이트)

: 등록이 발생하면 등록 가능 여부를 확인하고 상태 출력 및 등록 시, 대기순위를 자동으로 업데이트하는 기능을 수행한다.

① SQL CODE

```
19      CASE #이미 다른 유치원에 등록을 했는지 check
20          WHEN (SELECT SUM(pk.enroll_check)
21                  FROM parents_has_kindergarten pk
22                  where pk.parents_parent_id = parentsid) >= 1
23          THEN select concat("이미 등록된 유치원이 존재합니다: ", (select pk.kindergarten_kindergarten_id
24                                                                    from parents_has_kindergarten pk
25                                                                    where pk.parents_parent_id = parentsid and pk.enroll_check = 1) ) as result;
```

이미 다른 유치원에 등록을 했는지를,
Parents_parent_id table에 접근하여 확인하고
등록했을 경우, 등록됐다는 string을 출력

```
26          #등록절차
27          WHEN wait_rank <= 0
28          THEN
29              begin
30              update parents_has_kindergarten
31              set enroll_check = 1 #등록처리
32              where (enroll_check = 0 and parents_parent_id = parentsid and kindergarten_kindergarten_id = kinderid);
```

Wait_rank가 0 이하일 경우, 해당 유치원에 등록 가능한 상태.
해당 CASE의 경우 **등록 절차를 진행**함.
첫번째로, 해당 parentsid, kinderid에 해당하는 지원 정보의
enroll_check 열을 1로 바꿔 등록 처리함.

7 시스템 운영 Procedure

'Enrollkinder' Procedure (등록 및 대기순위 업데이트)

: 등록이 발생하면 등록 가능 여부를 확인하고 상태 출력 및 등록 시, 대기순위를 자동으로 업데이트하는 기능을 수행한다.

① SQL CODE

```
34              #대기 순위 업데이트
35              update parents_has_kindergarten
36              set now_rank = greatest(now_rank-1 , 0)
37              where kindergarten_kindergarten_id in (select t.forgive from temp_forgive t);
38
39              #상태 출력
40              select concat("등록처리 되었습니다: ", kinderid) as result;
41              end;
42
43          ELSE select concat("등록불가. 대기 순위를 확인하세요: ", wait_rank) as result ;
44      END CASE ;
45  end; //
46  delimiter ;
```

등록 절차 진행 시, 지원하였으나 등록하지 않은 유치원들에 자동으로 등록포기 처리됨.
따라서 **등록 포기 처리된 유치원**들의 대기 순위에 해당 인원이 빠져 **업데이트가 필요**함.
Temp_forgive에서 등록포기한 유치원 목록을 참조해 now_rank를 업데이트.

7 시스템 운영 Procedure

'Enrollkinder' Procedure (등록 및 대기순위 업데이트)

: 등록이 발생하면 등록 가능 여부를 확인하고 상태 출력 및 등록 시, 대기순위를 자동으로 업데이트하는 기능을 수행한다.

② 실행 결과

```
57 • call Enrollkinder('kid045','kind001');
58 • call Enrollkinder('kid045','kind005');
59 • call Enrollkinder('kid045','kind009');
```

result
등록불가. 대기 순위를 확인하세요: 3

result
등록처리 되었습니다: kind005

result
이미 등록된 유치원이 존재합니다: kind005

등록이 가능한지 여부에 따라 상태를 출력함. 그리고 등록 시, 등록과 대기순위 업데이트를 수행

```
50  #before
51  SET sql_safe_updates = 0; #일시적으로 safe update 해제
52 • select * from parents_has_kindergarten pk where pk.parents_parent_id = 'kid045';
53 • select * from parents_has_kindergarten pk where pk.kindergarten_kindergarten_id = 'kind001';
```

parents_parent_id	kindergarten_kindergarten_id	apply_rank	now_rank	enroll_check
kid045	kind001	3	3	0
kid045	kind005	0	0	0
kid045	kind009	0	0	0
NULL	NULL	NULL	NULL	NULL

parents_parent_id	kindergarten_kindergarten_id	apply_rank	now_rank	enroll_check
kid001	kind001	50	50	0
kid005	kind001	30	30	0
kid009	kind001	0	0	1
kid013	kind001	0	0	1
kid017	kind001	0	0	1

db변화

```
61  #after
62 • select * from parents_has_kindergarten pk where pk.parents_parent_id = 'kid045';
63 • select * from parents_has_kindergarten pk where pk.kindergarten_kindergarten_id = 'kind001';
```

parents_parent_id	kindergarten_kindergarten_id	apply_rank	now_rank	enroll_check
kid045	kind001	3	2	0
kid045	kind005	0	0	1
kid045	kind009	0	0	0
NULL	NULL	NULL	NULL	NULL

등록

parents_parent_id	kindergarten_kindergarten_id	apply_rank	now_rank	enroll_check
kid001	kind001	50	49	0
kid005	kind001	30	29	0
kid009	kind001	0	0	1
kid013	kind001	0	0	1

대기순위 업데이트

7 시스템 운영 Procedure

‘update_enrolled’ Procedure (등록 인원 업데이트)

:등록 학부모 수를 업데이트 하는 프로시저. 유치원 table 내에 해당 유치원의 ‘등록 학부모수’ cell의 숫자를 업데이트 한다.

① SQL CODE

```
1  DROP PROCEDURE IF EXISTS update_enrolled;
2  DELIMITER $$
3  CREATE PROCEDURE update_enrolled(IN enrolled_kinder_id VARCHAR(30))#, OUT enrolled_parents_sum INT)
4  BEGIN
5      UPDATE kindergarten
6          SET enroll_parents = (SELECT SUM(enroll_check) FROM parents_has_kindergarten
7                                WHERE kindergarten_kindergarten_id = enrolled_kinder_id)
8          WHERE kindergarten.kindergarten_id = enrolled_kinder_id;
9  END $$
10 DELIMITER ;
11 SET sql_safe_updates = 0; #일시적으로 safe update 해제
12 CALL update_enrolled("kind001")
```

8 Query 1 수행 및 결과

(USER : 학부모, 기간 : 선발 전)

Q1. 학부모 A의 선호 조건에 적합하는 유치원 목록은 무엇인가?

✓ Decision table 생성

establishtype	caretype	bus	type_id
public	standard	o	A
public	all	o	B
public	night	o	C
public	morning	o	D
private	standard	o	E
private	all	o	F
private	night	o	G

설립유형(국공립, 사립), 돌봄유형(기본운영, 온종일, 저녁, 아침), 통학차량 유무 등의 기준을 통해 유치원을 16가지로 분류해 DB 내 table 생성

① SQL CODE

```
select p.parent_id, p.wanttype, k.kindergarten_id
from parents p, kindergarten k
where  p.parent_id = 'kid001' and p.wanttype = k.type;
```

학부모 table에서 학부모가 원하는 조건에 따라 분류된 선호 type과 같은 type을 가진 유치원 ID를 유치원 table에서 가져와 조회

② 실행 결과

parent_id	wanttype	kindergarten_id
kid001	A	kind001
kid001	A	kind005
kid001	A	kind009

8 Query 2 수행 및 결과

(USER : 학부모, 기간 : 선발 전)

Q2. 유치원 B의 경쟁률 및 선호율은 어떻게 되는가?

① SQL CODE

```sql
#전체 유치원 경쟁률 확인하기
select k.name as '유치원',  k.likes/k.headcount as '경쟁률(배수)' ,
k.likes/(select count(*) from parents)*100 as '선호율(%)'
from kindergarten as k
order by likes DESC;

#특정 유치원 'water'에 대한 경쟁률 확인하기
select k.name as '유치원',  k.likes/k.headcount as '경쟁률(배수)' ,
k.likes/(select count(*) from parents)*100 as '선호율(%)'
from kindergarten k
where k.name = 'water';
```

유치원 Table에서 관심수(likes)와 정원 값(headcount)를 사용해 경쟁률과 선호율을 계산하고 이를 select하여 결과 column으로 조회

② 실행 결과

유치원	경쟁률(배수)	선호율(%)
love	4.4444	86.9565
meme	4.2857	65.2174
pen	8.3333	54.3478
cup	5.0000	54.3478
book	2.5000	21.7391
water	2.0000	21.7391
rice	2.0000	17.3913
aunt	2.6667	17.3913
apple	4.0000	17.3913

유치원	경쟁률(배수)	선호율(%)
water	2.0000	21.7391

8 Query 3 수행 및 결과

(USER : 학부모, 기간 : 선발 후~등록기간 종료 전)

Q3. 내가 '대기 상태'인 유치원에서의 내 대기 순위는 얼마인가?

① SQL CODE

```
2     #학부모의 대기 순위 및 상태 조회 프로시저
3     drop procedure if exists Myrank;
4     delimiter //
5     create procedure Myrank(
6     IN parentsid varchar(45))
7
8     #등록을 마쳤다면 등록을 했다고 안내해주기
9     If (select sum(pk.enroll_check) from parents_has_kindergarten pk
10        where pk.parents_parent_id = parentsid) >= 1
11    then begin
12        select "이미 등록된 유치원이 존재합니다" as result;
13    end;
15    #그렇지 않을 경우 해당 지원자의 순위 출력해서 보여주기
16    ELSE begin
17        select pk.kindergarten_kindergarten_id as '유치원',
18            pk.apply_rank as '최초순위' , pk.now_rank as '현재순위'
19        from parents_has_kindergarten as pk
20        where pk.parents_parent_id = parentsid;
21    end;
22    end if; //
23    delimiter ;
```

<'Myrank' procedure>

- 학부모ID 입력 시, 지원한 유치원에서의 대기 순위 조회
- 학부모ID를 input으로 입력 시, 등록을 하지 않은 학부모를 대상으로 지원한 유치원들의 최초 순위와 현재 순위를 출력
- 등록을 마친 학부모의 경우, 이미 등록된 유치원이 존재한다는 결과 메시지가 출력

② 실행 결과

```
25    call Myrank('kid046');
26    call Myrank('kid032');
```

유치원	최초순위	현재순위
kind004	3	3

→ [아직 **등록을 하지 않은** 학부모가 검색]
+ 'kid046'은 kind004에만 지원함

result
이미 등록된 유치원이 존재합니다

→ [이미 **등록을 한** 학부모가 검색]

8 Query 4 수행 및 결과

(USER : 학부모, 기간 : 선발 후~등록기간 종료 전)

Q4. 유치원 B에서 결원이 발생했는가?

① SQL CODE

```
DROP PROCEDURE IF EXISTS vacancy;
DELIMITER $$
CREATE PROCEDURE vacancy(IN kinder_name VARCHAR(30), OUT vacancy_status INT)
BEGIN
    DECLARE head INT;
    DECLARE enrolled INT;
    SELECT headcount INTO head FROM kindergarten WHERE name = kinder_name;
    SELECT enroll_parents INTO enrolled FROM kindergarten WHERE name = kinder_name;
    IF (head-enrolled) > 0 THEN SET vacancy_status = head-enrolled;
    ELSE SET vacancy_status = 0;
    END IF;
END $$
DELIMITER ;
CALL vacancy("love", @vacancy);
SELECT CONCAT("kindergarten named 'love' has ", @vacancy, " vacancies") as VACANCY_STATUS;
```

<'vacancy' procedure>

- 입력 받은 유치원 이름의 공석을 확인하는 procedure
- Procedure 실행 시, '정원 – 현재 등록 인원'(공석) 을 받아옴

② 실행 결과

VACANCY_STATUS
kindergarten named 'love' has 9 vacancies

8 Query 5 수행 및 결과

(USER : 유치원, 기간 : 선발 후~등록기간 종료 전)

Q5. 유치원 B에 대기 중인 학부모 수는 몇인가?

① SQL CODE

```
select k.name, h.kindergarten_kindergarten_id, max(h.now_rank) as '대기중인인원'
from kindergarten k, parents_has_kindergarten h
where k.kindergarten_id=h.kindergarten_kindergarten_id and h.kindergarten_kindergarten_id='kind001';
```

- 최초 선발 이후, 특정 유치원의 현재 대기 중인 인원을 알려줌
- 특정 유치원ID 입력 시, 유치원 table에서 해당 유치원의 현재대기순위 중 가장 높은 수(마지막 순위)를 '대기중인 인원' 결과 column으로 조회

② 실행 결과 : 유치원 ID 'kind001' 검색 결과

name	kindergarten_kindergarten_id	대기중인인원
love	kind001	2

8 Query 6 수행 및 결과

(USER : 유치원, 기간 : 등록기간 후 ~ 추가모집 마감)

Q6. 유치원 B의 정원보다 등록 인원이 적은가?

① SQL CODE

```
select kindergarten.kindergarten_id as '유치원 ID', kindergarten.name,
if (sum(parents_has_kindergarten.enroll_check) < kindergarten.headcount,
'등록 기간동안 미달', '등록 기간동안 정원을 충족함') as 등록여부
from parents_has_kindergarten, kindergarten
where  parents_has_kindergarten.kindergarten_kindergarten_id = 'kind001'
group by parents_has_kindergarten.kindergarten_kindergarten_id;
```

② 실행 결과 : 유치원 ID 'kind001' 검색 결과

유치원 ID	name	등록여부
kind001	love	등록 기간동안 정원을 충족함

- 등록기간 후, 어린이집ID 입력 시 각 어린이집의 미달 여부 확인 가능
- 지원 table에서 '등록여부'의 합과 유치원 table에서 '정원' 값을 비교하여 '등록여부' 결과 column에서 결과 메시지를 통해 미달, 충족 사항 조회

8 Query 7 수행 및 결과

(USER : 시스템 관리자)

Q7. 현재는 유치원 모집 일정의 어느 단계인가?

① SQL CODE

```
select
  case
  when start_date <=str_to_date(NOW(), '%-%-%')<=result_date
  then '유치원 지원기간입니다. 선발 결과는 2023-05-08에 나옵니다.'
  when result_date<=str_to_date(NOW(), '%-%-%')<=add_date
  then '선발 결과가 발표되었습니다. 등록/포기 기간입니다.'
  when str_to_date(NOW(), '%-%-%')=add_result_date
  then '등록/포기 기간이 지났습니다. 추가모집기간입니다.'
  else '유치원 지원기간이 아닙니다.'
  end as result
FROM mydb.system;
```

- 현재 시간과 비교해 유치원 모집 일정 단계 조회 가능
- 시스템 관리자 table의 '선발시작날짜', '선발결과발표날짜', '추가모집시작날짜'와 현재 시간을 비교해 해당되는 조건에 맞는 결과를 'result' 결과 column에 출력

② 실행 결과 : 유치원 ID 'kind001' 검색 결과

result
유치원 지원기간이 아닙니다.

모집 일정을 2023년 가정으로 했기에,
'유치원 지원기간이 아닙니다.' 결과가 조회됨.


#
Thanks!