

1. Цель работы: По аналогии с примерами, приведенными в п. 1 , создать в БД ХП, реализующие: — вставку с пополнением справочников (вставляется информация о студенте, если указанный номер группы отсутствует в БД, запись добавляется в таблицу с перечнем групп) (получаем ссылку на внешний ключ по значению данного из родительской таблицы); — удаление с очисткой справочников – удаление всех зависимых данных(удаляется информация о студенте, если в его группе нет больше студентов, запись удаляется из таблицы с перечнем групп); — каскадное удаление (перед удалением записи о группе удаляются записи обо всех студентах этой группы); — вычисление и возврат значения агрегатной функции (т.к. агрегатная функция дает единственный результат); — формирование статистики во временной таблице (например, для рассматриваемой БД — для каждого факультета: количество групп, количество обучающихся студентов, количество изучаемых дисциплин, количество сданных дисциплин, средний балл по факультету).

2. Задание на лабораторную работу.

9. охраняемые парковки: адрес парковки, машина, владелец, место, рег. номер машины, дата и время заезда, дата и время выезда

а. все парковки, расположенные на улицах, в названии которых есть слово «Малая», но на него название не заканчивается

б. владелец машины, у которого несколько машин разных марок

в. улица, на которой нет парковок

г. парковка, открывающаяся позже всех

д. владелец машины, останавливавшийся на парковках, количество которых больше среднего

е. машина, которая стояла на всех парковках Центрального района

ж. владелец, не парковавшийся на Вознесенском проспекте, но парковавшийся в Московском районе

3. Физическая модель предметной области.


```

        brand_id_new := (SELECT coalesce(MAX(brand_id)+1, 0) FROM
brand);

        INSERT INTO brand(brand_id, brand_name)
        VALUES (brand_id_new, namebrand);

        END;

END IF;

car_id_new:=(SELECT coalesce(MAX(car_id)+1, 0) FROM car);
IF NOT EXISTS (SELECT * FROM car WHERE reg_num= numreg)
THEN INSERT INTO car(car_id, reg_num, fk_brand)
VALUES(car_id_new, numreg, brand_id_new);
END IF;

END;

$$ LANGUAGE plpgsql;

```

```
CALL ins_car('Honda', 'ш674ey');
```

```
CALL ins_car('Tesla', 'o840HH');
```

```
select * from car;
```

```
select * from brand;
```

```
-----
--удаление с очисткой справочников
```

```
DROP PROCEDURE del_car_clear_br(car_id_del int);
```

```
CREATE PROCEDURE del_car_clear_br(car_id_del int) AS $$
```

```
DECLARE fk_brand_del int;
```

```
BEGIN
```

```
SELECT fk_brand INTO fk_brand_del FROM car WHERE car_id =
car_id_del;
```

```
DELETE FROM car WHERE car_id = car_id_del;
```

```
IF NOT EXISTS(SELECT * FROM car WHERE fk_brand = fk_brand_del)
```

```
THEN DELETE FROM brand WHERE brand_id = fk_brand_del;
```

```
END IF;
```

END;

\$\$ LANGUAGE plpgsql;

CALL del_car_clear_br(2);

CALL del_car_clear_br(3);

select * from car;

select * from brand;

--каскадное удаление

DROP PROCEDURE del_brand_cascade(brand_id_del int);

CREATE PROCEDURE del_brand_cascade(brand_id_del int) AS \$\$

BEGIN

DELETE FROM car_parking

WHERE fk_car IN (SELECT car_id FROM car WHERE
fk_brand=brand_id_del);

DELETE FROM car_owner

WHERE fk_car IN (SELECT car_id FROM car WHERE
fk_brand=brand_id_del);

DELETE FROM car WHERE fk_brand=brand_id_del;

DELETE FROM brand WHERE brand_id=brand_id_del;

END;

\$\$ LANGUAGE plpgsql;

CALL del_brand_cascade(2);

select * from car;

select * from car_owner;

select * from car_parking;

```
select * from brand;
```

--вычисление и возврат значения агрегатной функции

```
DROP PROCEDURE count_brand_out(OUT count_brand int);  
CREATE OR REPLACE PROCEDURE count_brand_out(OUT  
count_brand int) AS $$  
BEGIN  
SELECT COALESCE(COUNT(brand_id), 0) INTO count_brand FROM  
brand;  
END;  
$$ LANGUAGE plpgsql;
```

```
CALL count_brand_out(null);
```

```
DROP FUNCTION count_brand_out_fun(OUT count_brand int);  
CREATE OR REPLACE FUNCTION count_brand_out_fun(OUT  
count_brand int) AS $$  
BEGIN  
SELECT COALESCE(COUNT(brand_id), 0) INTO count_brand FROM  
brand;  
END;  
$$ LANGUAGE plpgsql;
```

```
SELECT count_brand_out_fun();
```

--формирование статистики во временной таблице

```
DROP FUNCTION cas_statistics();  
CREATE OR REPLACE FUNCTION cas_statistics()  
RETURNS TABLE(id_c INT, o_count INT, parking_count INT,  
street_count INT, district_count INT) AS $$
```

BEGIN

-- Создаем временную таблицу для хранения статистики

CREATE TEMPORARY TABLE stat_table

(id_stat serial PRIMARY KEY,

id_car INT,

owner_count INT,

parking_count INT,

street_count INT,

district_count INT

);

-- Заполняем временную таблицу статистикой владельцев

INSERT INTO stat_table (id_car, owner_count)

SELECT

car.car_id,

COUNT(DISTINCT car_owner.fk_owner) AS owner_count

FROM

car

LEFT JOIN

car_owner ON car.car_id = car_owner.fk_car

GROUP BY

car.car_id;

-- Обновляем временную таблицу для учета парковок

UPDATE stat_table

SET

parking_count = COALESCE(subquery.parking_count, 0)

FROM (

SELECT

car.car_id AS sub_car_id,

```

COUNT(DISTINCT car_parking.fk_parking) AS parking_count
FROM
    car
LEFT JOIN
    car_parking ON car.car_id = car_parking.fk_car
GROUP BY
    sub_car_id
) AS subquery
WHERE
    stat_table.id_car = subquery.sub_car_id;

```

```

-- Обновляем временную таблицу для учета улиц
UPDATE stat_table
SET
    street_count = COALESCE(subquery.streets_count, 0)
FROM (
    SELECT
        car.car_id AS sub_car_id,
        COUNT(DISTINCT street.street_id) AS streets_count
    FROM
        car
    LEFT JOIN
        car_parking ON car.car_id = car_parking.fk_car
    LEFT JOIN
        parking ON car_parking.fk_parking = parking.parking_id
    LEFT JOIN
        street ON parking.fk_street = street.street_id
    GROUP BY
        sub_car_id
) AS subquery

```

WHERE

stat_table.id_car = subquery.sub_car_id;

-- Обновляем временную таблицу для учета районов

UPDATE stat_table

SET

district_count = COALESCE(subquery.district_count, 0)

FROM (

SELECT

car.car_id AS sub_car_id,

COUNT(DISTINCT district.district_id) AS district_count

FROM

car

LEFT JOIN

car_parking ON car.car_id = car_parking.fk_car

LEFT JOIN

parking ON car_parking.fk_parking = parking.parking_id

LEFT JOIN

district ON parking.fk_district = district.district_id

GROUP BY

sub_car_id

) AS subquery

WHERE

stat_table.id_car = subquery.sub_car_id;

-- Возвращаем результаты

RETURN QUERY SELECT id_car, owner_count,
stat_table.parking_count, stat_table.street_count, stat_table.district_count
FROM stat_table;


```
-- Очищаем временную таблицу
DROP TABLE IF EXISTS stat_table;




END;



$$ LANGUAGE plpgsql;
```

```
SELECT * FROM cas_statistics();
```

5. Наборы данных, возвращаемые запросами

Вставка с пополнением справочников До

	car_id [PK] integer 	reg_num character varying (10) 	fk_brand integer 
1	1	ш674еу	2
2	2	г123цу	1
3	3	з908дл	2
4	4	и236еп	3
5	5	ф825мп	5

	brand_id [PK] integer 	brand_name character varying (20) 
1	1	BMW
2	2	Honda
3	3	Volkswagen
4	4	Ford
5	5	Hyundai

После

	car_id [PK] integer	reg_num character varying (10)	fk_brand integer
1	1	ш674еу	2
2	2	г123цу	1
3	3	э908дл	2
4	4	и236еп	3
5	5	ф825мп	5
6	6	о840нн	6

	brand_id [PK] integer	brand_name character varying (20)
1	1	BMW
2	2	Honda
3	3	Volkswagen
4	4	Ford
5	5	Hyundai
6	6	Tesla

Рис. 1 вывод данных по хп вставка с пополнением справочников

Удаление с очисткой справочников

До

	car_id [PK] integer	reg_num character varying (10)	fk_brand integer
1	1	ш674еу	2
2	2	г123цу	1
3	3	э908дл	2
4	4	и236еп	3
5	5	ф825мп	5
6	6	о840нн	6

	brand_id [PK] integer	brand_name character varying (20)
1	1	BMW
2	2	Honda
3	3	Volkswagen
4	4	Ford
5	5	Hyundai
6	6	Tesla

После

	car_id [PK] integer	reg_num character varying (10)	fk_brand integer
1	1	ш674еу	2
2	4	и236ен	3
3	5	ф825мр	5
4	6	о840нн	6

	brand_id [PK] integer	brand_name character varying (20)
1	2	Honda
2	3	Volkswagen
3	4	Ford
4	5	Hyundai
5	6	Tesla

Рис. 2 вывод данных по хп удаление с очисткой справочников

Каскадное удаление

До

	brand_id [PK] integer	brand_name character varying (20)
1	2	Honda
2	3	Volkswagen
3	4	Ford
4	5	Hyundai
5	6	Tesla

	car_id [PK] integer	reg_num character varying (10)	fk_brand integer
1	1	ш674ey	2
2	4	и236en	3
3	5	ф825mp	5
4	6	о840нн	6

	car_owner_id [PK] integer	fk_car integer	fk_owner integer
1	1	1	2
2	4	4	5
3	5	5	1

	car_parking_id [PK] integer	arrival timestamp without time zone	parking_space character varying (5)	departure timestamp without time zone	fk_car integer	fk_parking integer
1	2	2023-10-18 10:30:20	B2	2023-10-18 16:20:45	1	5
2	4	2023-10-18 14:20:50	D4	2023-10-18 20:10:25	5	3
3	5	2023-10-18 16:55:05	E5	2023-10-18 22:35:15	1	2
4	6	2023-10-18 17:55:05	W5	2023-10-18 19:35:15	1	1

После

	brand_id [PK] integer	brand_name character varying (20)
1	3	Volkswagen
2	4	Ford
3	5	Hyundai
4	6	Tesla

	car_id [PK] integer	reg_num character varying (10)	fk_brand integer
1	4	и236еп	3
2	5	ф825мп	5
3	6	о840нн	6

	car_owner_id [PK] integer	fk_car integer	fk_owner integer
1	4	4	5
2	5	5	1

	car_parking_id [PK] integer	arrival timestamp without time zone	parking_space character varying (5)	departure timestamp without time zone	fk_car integer	fk_parking integer
1	4	2023-10-18 14:20:50	D4	2023-10-18 20:10:25	5	3

Рис. 3 вывод данных по хп каскадное удаление

Вычисление и возврат значения агрегатной функции

	count_brand integer
1	5

	count_brand_out_fun integer
1	5

Рис. 4 вывод данных по хп вычисление и возврат значения агрегатной функции

Формирование статистики во временной таблице






	id_c integer 	o_count integer 	parking_count integer 	street_count integer 	district_count integer 
1	1	1	3	3	2
2	2	1	1	1	1
3	3	1	1	1	1
4	4	1	0	0	0
5	5	1	1	1	1
6	6	0	0	0	0

Рис. 5 вывод данных по хп формирование статистики во временной
таблице