

2 Лабораторная работа №2

«Язык SQL. Генераторы. Триггеры»

2.1 Цель работы:

Ознакомиться с принципом работы генераторов и триггеров, продемонстрировать работу на примерах.

2.2 Постановка задачи

2.2.1 Записать запросы, соединяющие две таблицы с помощью JOIN и без него.

2.2.2 Записать запросы, соединяющие более двух таблиц с помощью JOIN и без него.

2.2.3 Продemonстрировать следующие возможности SQL:

- использование псевдонимов на примере рекурсивного запроса;
- привести пример запроса с подзапросом;
- использование агрегатных функций в подзапросе;
- подзапросы, возвращающие единственное и множественные значения;
- подзапросы, использующие вычисление;
- использование подзапросов в HAVING.

2.2.4 Записать запрос, соединяющий таблицу со своей копией.

2.2.5 Привести пример коррелированного запроса, использующего две разные таблицы.

2.2.6 Продemonстрировать следующие возможности SQL

- работу оператора EXISTS;
- работу оператора ALL;
- работу оператора ANY.

2.2.7 В соответствии с вариантом задания (Вариант – 5) создать генератор и триггер:

Создать генераторы для полей «Номер фирмы», «Номер типа помещения». Автоматическая генерация полей «Номер фирмы», «Номер типа помещения», создать обновляемое представление «ПомещениеТип» (в таблицу «Помещение» вместо номера типа помещения подставить наименование типа помещения).

2.2.8 Изменить значение генератора, в соответствии с хранимыми данными.

2.2.9 Ввести данные в таблицу, используя генератор (не менее 5 строк). Просмотреть полученный результат.

2.2.10 Внести изменения в указанные таблицы, используя триггеры (не менее 5 строк). Просмотреть полученный результат.

2.3 Ход работы

2.3.1 Было осуществлено подключение к базе данных из прошлой лабораторной работы с помощью следующего запроса:

```
CONNECT "D:\5_semester\Databases\Lab_1\Firms.fdb" user 'SYSDBA' password 'masterkey';
```

Был сделан запрос, соединяющий две таблицы без Join: “Вывести из таблицы Помещение атрибут номер фирмы, из таблицы Телефон номер телефона, Из таблиц Помещение и Телефон, Где в таблице Помещение атрибут адреса равен атрибуту адреса в таблице Телефон”. Результат выполнения запроса отображен на рисунке 2.1.

```
SELECT ROOM.COMPANY_NUM, TELEPHONE.PHONE_NUM  
FROM ROOM, TELEPHONE  
WHERE ROOM.ADRESS = TELEPHONE.ADRESS;
```

```
SQL> SELECT ROOM.COMPANY_NUM, TELEPHONE.PHONE_NUM  
CON> FROM ROOM, TELEPHONE  
CON> WHERE ROOM.ADRESS = TELEPHONE.ADRESS;  
  
COMPANY_NUM PHONE_NUM  
=====
```

280040223	+7 (917) 921-43-15
280040223	+7 (936) 293-61-29
447095648	+7 (953) 243-45-28
317286530	+7 (989) 785-16-63
280040223	+7 (957) 469-96-49
74792076	+7 (973) 622-62-47
74792076	+7 (998) 814-35-97
317286530	+7 (954) 675-54-12
447095648	+7 (990) 670-11-83
317286530	+7 (960) 893-55-81
447095648	<null>
74792076	<null>
280040223	+7 (913) 849-82-92
317286530	+7 (925) 618-64-30

Рисунок 2.1 – Результат запроса соединения двух таблиц без использования Join

Затем такой же запрос был сделан с использованием Join. Результат выполнения запроса отображен на рисунке 2.2. Запрос был следующий: “Вывести из таблицы Помещение атрибут номер фирмы, из таблицы Телефон номер телефона, Из таблиц Помещение соединяя с таблицей Телефон Где Помещение атрибут адреса равен атрибуту адреса в таблице Телефон”.

```
SELECT ROOM.COMPANY_NUM, TELEPHONE.PHONE_NUM  
FROM ROOM JOIN TELEPHONE  
ON ROOM.ADRESS = TELEPHONE.ADRESS;
```

```
SQL> SELECT ROOM.COMPANY_NUM, TELEPHONE.PHONE_NUM  
CON> FROM ROOM JOIN TELEPHONE  
CON> ON ROOM.ADRESS = TELEPHONE.ADRESS;  
  
COMPANY_NUM PHONE_NUM  
=====
```

280040223	+7 (917) 921-43-15
280040223	+7 (936) 293-61-29
447095648	+7 (953) 243-45-28
317286530	+7 (989) 785-16-63
280040223	+7 (957) 469-96-49
74792076	+7 (973) 622-62-47
74792076	+7 (998) 814-35-97
317286530	+7 (954) 675-54-12
447095648	+7 (990) 670-11-83
317286530	+7 (960) 893-55-81
447095648	<null>
74792076	<null>
280040223	+7 (913) 849-82-92
317286530	+7 (925) 618-64-30

Рисунок 2.2 – Результат запроса соединения двух таблиц с использованием Join

2.3.2 Были соединены три таблицы сперва без использования Join и результат выполнения запроса отображен на рисунке 2.3. Запрос был следующий: “Вывести на экран из таблицы Помещение атрибут номер фирмы, из таблицы Тип помещения атрибут тип помещения, из таблицы Телефон атрибут номер телефона Из таблиц Помещение, Телефон, Тип помещения Где адрес из таблицы Помещение равен адресу из таблицы Телефон И где номер типа помещения в таблице Тип помещения равен номеру типа помещения из таблицы Помещение”.

```
SELECT ROOM.COMPANY_NUM, ROOM_TYPE.ROOM_TYPE, TELEPHONE.PHONE_NUM  
FROM ROOM, TELEPHONE, ROOM_TYPE  
WHERE ROOM.ADRESS = TELEPHONE.ADRESS AND  
ROOM_TYPE.ROOM_TYPE_NUM = ROOM.ROOM_TYPE_NUM;
```

```
SQL> SELECT ROOM.COMPANY_NUM, ROOM_TYPE.ROOM_TYPE, TELEPHONE.PHONE_NUM
CON> FROM ROOM, TELEPHONE, ROOM_TYPE
CON> WHERE ROOM.ADRRESS = TELEPHONE.ADRRESS AND
CON> ROOM_TYPE.ROOM_TYPE_NUM = ROOM.ROOM_TYPE_NUM;
```

COMPANY_NUM	ROOM_TYPE	PHONE_NUM
280040223	Couloir	+7 (917) 921-43-15
74792076	Workshop	+7 (998) 814-35-97
280040223	Workshop	+7 (913) 849-82-92
280040223	Shop	+7 (957) 469-96-49
447095648	Office	+7 (990) 670-11-83
74792076	Office	<null>
317286530	Warehouse	+7 (954) 675-54-12
317286530	Warehouse	+7 (960) 893-55-81
280040223	Technical under...	+7 (936) 293-61-29
447095648	Technical under...	+7 (953) 243-45-28
317286530	Technical under...	+7 (989) 785-16-63
317286530	Technical under...	+7 (925) 618-64-30
74792076	<null>	+7 (973) 622-62-47

Рисунок 2.3 – Результат запроса соединения трёх таблиц без использования Join

Затем были соединены три таблицы сперва с использования Join и результат выполнения запроса отображен на рисунке 2.4. Запрос был следующий: “Вывести на экран из таблицы Помещение атрибут номер фирмы, из таблицы Тип помещения атрибут тип помещения, из таблицы Телефон атрибут номер телефона Из таблиц (Помещение соединяя с таблицей Телефон, где адрес из таблицы Помещение равен адресу из таблицы Телефон) Соединяя с таблицей Тип помещения, где номер типа помещения в таблице Тип помещения равен номеру типа помещения из таблицы Помещение Сортируя по номеру фирмы из таблицы Помещение”.

```
SELECT ROOM.COMPANY_NUM, ROOM_TYPE.ROOM_TYPE, TELEPHONE.PHONE_NUM
FROM (ROOM JOIN TELEPHONE ON ROOM.ADRRESS = TELEPHONE.ADRRESS)
JOIN ROOM_TYPE ON ROOM_TYPE.ROOM_TYPE_NUM = ROOM.ROOM_TYPE_NUM
ORDER BY ROOM.COMPANY_NUM;
```

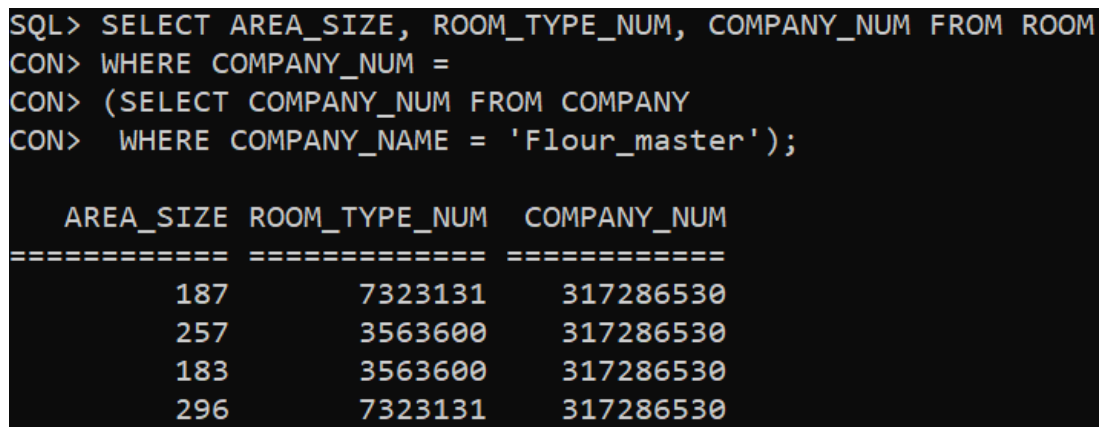
```
SQL> SELECT ROOM.COMPANY_NUM, ROOM_TYPE.ROOM_TYPE, TELEPHONE.PHONE_NUM
CON> FROM (ROOM JOIN TELEPHONE ON ROOM.ADRRESS = TELEPHONE.ADRRESS)
CON> JOIN ROOM_TYPE ON ROOM_TYPE.ROOM_TYPE_NUM = ROOM.ROOM_TYPE_NUM
CON> ORDER BY ROOM.COMPANY_NUM;
```

COMPANY_NUM	ROOM_TYPE	PHONE_NUM
74792076	Workshop	+7 (998) 814-35-97
74792076	<null>	+7 (973) 622-62-47
74792076	Office	<null>
280040223	Shop	+7 (957) 469-96-49
280040223	Couloir	+7 (917) 921-43-15
280040223	Workshop	+7 (913) 849-82-92
280040223	Technical under...	+7 (936) 293-61-29
317286530	Warehouse	+7 (954) 675-54-12
317286530	Warehouse	+7 (960) 893-55-81
317286530	Technical under...	+7 (925) 618-64-30
317286530	Technical under...	+7 (989) 785-16-63
447095648	Office	+7 (990) 670-11-83
447095648	Technical under...	+7 (953) 243-45-28

Рисунок 2.4 – Результат запроса соединения трёх таблиц с использованием Join

2.3.3 Был приведен пример запроса с подзапросом, содержащий следующую смысловую нагрузку: “Вывести на экран Размер помещения, Номер типа помещения, Номер фирмы из таблицы Помещение, где Номер фирмы равен (номеру фирмы из таблицы Фирма, где имя компании = «Мучной мастер»)”. Результат выполнения запроса отображен на рисунке 2.5.

```
SELECT AREA_SIZE, ROOM_TYPE_NUM, COMPANY_NUM FROM ROOM
WHERE COMPANY_NUM =
    (SELECT COMPANY_NUM FROM COMPANY
    WHERE COMPANY_NAME = 'Flour_master');
```



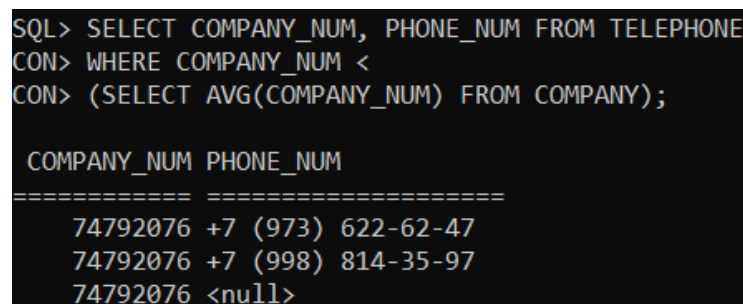
```
SQL> SELECT AREA_SIZE, ROOM_TYPE_NUM, COMPANY_NUM FROM ROOM
CON> WHERE COMPANY_NUM =
CON> (SELECT COMPANY_NUM FROM COMPANY
CON> WHERE COMPANY_NAME = 'Flour_master');
```

AREA_SIZE	ROOM_TYPE_NUM	COMPANY_NUM
187	7323131	317286530
257	3563600	317286530
183	3563600	317286530
296	7323131	317286530

Рисунок 2.5 – Результат выполнения запроса с подзапросом

Для демонстрации использования агрегатных функций в подзапросах был сделан запрос означающий: “Вывести на экран номер фирмы, номер телефона из таблицы Телефон, где номер фирмы меньше среднего значения номера фирмы из таблицы Фирма”. Результат отображения запроса отображен на рисунке 2.6.

```
SELECT COMPANY_NUM, PHONE_NUM FROM TELEPHONE
WHERE COMPANY_NUM <
    (SELECT AVG(COMPANY_NUM) FROM COMPANY);
```



```
SQL> SELECT COMPANY_NUM, PHONE_NUM FROM TELEPHONE
CON> WHERE COMPANY_NUM <
CON> (SELECT AVG(COMPANY_NUM) FROM COMPANY);
```

COMPANY_NUM	PHONE_NUM
74792076	+7 (973) 622-62-47
74792076	+7 (998) 814-35-97
74792076	<null>

Рисунок 2.6 – Результат выполнения запроса с подзапросом с использованием агрегатной функции

Для демонстрации работы подзапроса возвращающего множественное значение был совершен запрос означающий: “Выбрать все атрибуты из таблицы Помещение, где номер типа помещения в таблице тип помещения не равен null”. Запрос выдал ожидаемый результат и отображен на рисунке 2.7.

```
SELECT * FROM ROOM
WHERE ROOM_TYPE_NUM IN
(SELECT ROOM_TYPE_NUM FROM ROOM_TYPE
WHERE ROOM_TYPE <> 'null');
```

```
SQL> SELECT * FROM ROOM
CON> WHERE ROOM_TYPE_NUM IN
CON> (SELECT ROOM_TYPE_NUM FROM ROOM_TYPE
CON> WHERE ROOM_TYPE <> 'null');
```

ADRESS	COMPANY_NUM	AREA_SIZE	ROOM_TYPE_NUM
129296, Orel region, city of Mytishchi, Chekhov highway, 76	280040223	159	2983562
314799, Ulyanovsk region, Pushkino city, Domodedovo Boulevard, 05	280040223	247	7323131
546499, Arkhangelsk region, Taldom city, Ladygin highway, 73	447095648	116	7323131
733171, Bryansk region, Shatura city, Ladygin descent, 33	317286530	187	7323131
753258, Volgograd region, the city of Pavlovsky Posad, Slava passage, 52	280040223	245	5442937
211738, Novosibirsk region, the city of Voskresensk, highway Bucharest, 30	74792076	6	8897460
577097, Magadan region, the city of Silver Ponds, Gagarin Square, 63	317286530	257	3563600
022652, Chita region, Ramenskoye city, Bucharest entrance, 17	447095648	204	9077519
766338, Magadan region, Podolsk, Lomonosov str., 24	317286530	183	3563600
588852, Orenburg region, Domodedovo city, 1905 boulevard, 72	74792076	67	9077519
248314, Gogol Ave., Krasnogorsk, Chelyabinsk Region 71	280040223	157	8897460
105518, Sakhalin region, Dorokhovo city, lane. Lomonosov, 22	317286530	296	7323131

Рисунок 2.7 – Результат подзапроса, использующего множественное значение

Для демонстрации работы подзапросов, использующих вычисление был сделан запрос, означающий следующую информацию: “Вывести на экран номер фирмы, номер типа помещения из таблицы Помещение, где размер помещения должен быть меньше чем, номер типа помещения деленный на 100000 и где тип помещения офис”. Результат выполнения запроса отображен на рисунке 2.8.

```
SELECT COMPANY_NUM, ROOM_TYPE_NUM FROM ROOM
WHERE AREA_SIZE <
(SELECT ROOM_TYPE_NUM/100000 FROM ROOM_TYPE
WHERE ROOM_TYPE = 'Office');
```

```
SQL> SELECT COMPANY_NUM, ROOM_TYPE_NUM FROM ROOM
CON> WHERE AREA_SIZE <
CON> (SELECT ROOM_TYPE_NUM/100000 FROM ROOM_TYPE
CON> WHERE ROOM_TYPE = 'Office');
```

COMPANY_NUM	ROOM_TYPE_NUM
74792076	8897460
74792076	9077519

Рисунок 2.8 – Результат выполнения подзапроса с использованием вычисления

Наконец для демонстрации работы Having в подзапросах был сделан запрос означающий: “Вывести на экран номер фирмы, среднее значение размера помещения для этого номера фирмы, это среднее значение будет называться AVG_AREA_SIZE, все данные из таблицы Помещение, группируя по номеру фирмы, группировать только те значения где среднее значение размера помещения больше среднего значения помещения в таблице Помещение где номер фирмы не равен 74792076”. Результат выполнения запроса отображен на рисунке 2.9.

```
SELECT COMPANY_NUM, AVG(AREA_SIZE) AS AVG_AREA_SIZE FROM ROOM
GROUP BY COMPANY_NUM
HAVING AVG(AREA_SIZE) >
    (SELECT AVG(AREA_SIZE) FROM ROOM
     WHERE COMPANY_NUM <> 74792076);
```

```
SQL> SELECT COMPANY_NUM, AVG(AREA_SIZE) AS AVG_AREA_SIZE FROM ROOM
CON> GROUP BY COMPANY_NUM
CON> HAVING AVG(AREA_SIZE) >
CON> (SELECT AVG(AREA_SIZE) FROM ROOM
CON> WHERE COMPANY_NUM <> 74792076);
```

COMPANY_NUM	AVG_AREA_SIZE
317286530	230

Рисунок 2.9 – Результат выполнения подзапроса с использованием Having

2.3.4 Сделан запрос, выводящий номера фирм, номера типов помещения и площадь помещения из таблицы помещение, где площадь помещения выше средней по таблице. Результат выполнения запроса отображен на рисунке 2.10.

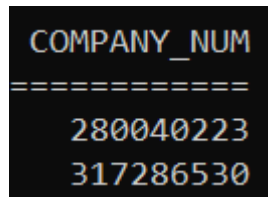
```
SELECT COMPANY_NUM, ROOM_TYPE_NUM, AREA_SIZE
FROM ROOM
WHERE AREA_SIZE >=
    (SELECT AVG(AREA_SIZE)
     FROM ROOM);
```

COMPANY_NUM	ROOM_TYPE_NUM	AREA_SIZE
280040223	7323131	247
317286530	7323131	187
280040223	5442937	245
317286530	3563600	257
447095648	9077519	204
317286530	3563600	183
447095648	<null>	193
317286530	7323131	296

Рисунок 2.10 – Результат запроса, соединяющего таблицу со своей копией

2.3.5 Для демонстрации работы коррелированного запроса с двумя таблицами был сделан запрос, несущий в себе следующий смысл: “Вывести номера фирм, имеющие больше трёх помещений”. Результат выполнения запроса соответствует ожиданиям и отображен на рисунке 2.11.

```
SELECT COMPANY_NUM FROM COMPANY main
WHERE 3 <
      (SELECT COUNT (AREA_SIZE) FROM ROOM
       WHERE main.COMPANY_NUM = COMPANY_NUM);
```



The screenshot shows a terminal window with a dark background. It displays the header 'COMPANY_NUM' followed by a line of equals signs. Below this, two company numbers are listed: '280040223' and '317286530'.

COMPANY_NUM
=====
280040223
317286530

Рисунок 2.11 – Результат выполнения, коррелированного запроса с двумя таблицами

2.3.6 Были созданы запросы с использованием EXISTS, ALL и ANY для демонстрации их работы в языке SQL. Результаты выполнения запросов отображены на рисунках 2.12 – 2.14.

Вывести все значения из таблицы Телефон Где не существует компаний с помещениями площадью меньше 100.

```
SELECT * FROM TELEPHONE main
WHERE EXIST
      (SELECT COMPANY_NUM FROM ROOM
       WHERE AREA_SIZE < 100 AND main.COMPANY_NUM <> COMPANY_NUM);
```

Выбрать все записи из таблицы телефон, где номер фирмы равен какому-либо номеру фирмы из таблицы Помещение, где размер помещения меньше 100.

```
SELECT * FROM TELEPHONE main
WHERE COMPANY_NUM = ANY
      (SELECT COMPANY_NUM FROM ROOM
       WHERE AREA_SIZE < 100);
```

Вывести кортежи из таблицы Помещение, где размер помещения больше, чем любой другой в этой фирме.

```
SELECT * FROM ROOM main
WHERE AREA_SIZE >= ALL (SELECT AREA_SIZE FROM ROOM WHERE main.COMPANY_NUM = COMPANY_NUM);
```


ADRESS	COMPANY_NUM	PHONE_NUM
129296, Orel region, city of Mytishchi, Chekhov highway, 76	280040223	+7 (917) 921-43-15
314799, Ulyanovsk region, Pushkino city, Domodedovo Boulevard, 05	280040223	+7 (936) 293-61-29
546499, Arkhangelsk region, Taldom city, Ladygin highway, 73	447095648	+7 (953) 243-45-28
733171, Bryansk region, Shatura city, Ladygin descent, 33	317286530	+7 (989) 785-16-63
753258, Volgograd region, the city of Pavlovsky Posad, Slava passage, 52	280040223	+7 (957) 469-96-49
577097, Magadan region, the city of Silver Ponds, Gagarin Square, 63	317286530	+7 (954) 675-54-12
022652, Chita region, Ramenskoye city, Bucharest entrance, 17	447095648	+7 (990) 670-11-83
766338, Magadan region, Podolsk, Lomonosov str., 24	317286530	+7 (960) 893-55-81
860921, Tver region, Domodedovo city, Cosmonauts Square, 91	447095648	<null>
248314, Gogol Ave., Krasnogorsk, Chelyabinsk Region 71	280040223	+7 (913) 849-82-92
105518, Sakhalin region, Dorokhovo city, lane. Lomonosov, 22	317286530	+7 (925) 618-64-30

Рисунок 2.12 – Демонстрация работы оператора EXISTS

ADRESS	COMPANY_NUM	PHONE_NUM
485934, Arkhangelsk region, Sergiev Posad, Stalins entrance, 07	74792076	+7 (973) 622-62-47
211738, Novosibirsk region, the city of Voskresensk, highway Bucharest, 30	74792076	+7 (998) 814-35-97
588852, Orenburg region, Domodedovo city, 1905 boulevard, 72	74792076	<null>

Рисунок 2.13 – Демонстрация работы оператора ANY

ADRESS	COMPANY_NUM	AREA_SIZE	ROOM_TYPE_NUM
314799, Ulyanovsk region, Pushkino city, Domodedovo Boulevard, 05	280040223	247	7323131
485934, Arkhangelsk region, Sergiev Posad, Stalins entrance, 07	74792076	167	1473798
022652, Chita region, Ramenskoye city, Bucharest entrance, 17	447095648	204	9077519
105518, Sakhalin region, Dorokhovo city, lane. Lomonosov, 22	317286530	296	7323131

Рисунок 2.14 – Демонстрация работы оператора ALL

2.3.7 Были созданы генераторы для Номера фирмы и Номера типа помещения и установлены для них начальные значения.

```
CREATE GENERATOR COMPANY_NUM_GEN;
SET GENERATOR COMPANY_NUM_GEN TO 1;
CREATE GENERATOR ROOM_TYPE_NUM_GEN;
SET GENERATOR ROOM_TYPE_NUM_GEN TO 10;
```

Создано представление:

```
CREATE VIEW RoomType AS
SELECT ADRESS, COMPANY_NUM, AREA_SIZE, ROOM_TYPE
FROM ROOM, ROOM_TYPE
WHERE ROOM.ROOM_TYPE_NUM = ROOM_TYPE.ROOM_TYPE_NUM;
```

Был создан триггер, который делает то, что от него хотят.

```
SET TERM !! ;
CREATE TRIGGER RoomType_Insert2 FOR RoomType BEFORE INSERT AS
BEGIN
INSERT INTO ROOM values (NEW.ADRESS, NEW.COMPANY_NUM, NEW.AREA_SIZE,
GEN_ID(ROOM_TYPE_NUM_GEN,1));
END !!
SET TERM ; !!
```

По правде говоря, я не понял, чего от меня хотят, чтобы я сделал. Хотелось бы чтобы постановка задачи формулировалась яснее. А так я пока могу лишь гадать что от меня хотят. Вставляю рисунок, демонстрирующий, что триггер всё же у меня написался без возникновения ошибок (Рисунок 2.15) и пожалуй на сегодня закончим.

```
SET TERM !! ;
CREATE TRIGGER RoomType_Insert2 FOR RoomType BEFORE INSERT AS
BEGIN
INSERT INTO ROOM values (NEW.ADRESS, NEW.COMPANY_NUM, NEW.AREA_SIZE, GEN_ID(ROOM_TYPE_NUM_GEN,1));
END !!
SET TERM ; !!
```

Рисунок 2.15 – Создание тригера

Выводы

При выполнении данной лабораторной работы были получены навыки работы с коррелированными и вложенными подзапросами, соединения двух и более таблиц с использованием и без использования Join. Были изучены соотнесения таблицы со своей копией. На практике были изучены и применены операторы EXISTS, ANY, ALL. Были изучены рекурсивные объединения и псевдонимы. Изучены и закрепились на практике вложенные подзапросы с использованием агрегатных функций, вычислений, множественных значений. Ознакомились с принципом работы генераторов и триггеров. Полученные знания и опыт помогут более комфортно, качественно и быстро работать с базами данных в будущем с помощью языка запросов SQL.

ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назначение генераторов?

Генератор – объект базы данных, служащий для генерации последовательностей целых чисел.

2. Как сгенерировать следующее значение генератора?

Для того чтобы получить следующее значение генератора, необходимо применить функцию GEN_ID (здесь name – имя генератора, step – инкремент генератора):

```
GEN_ID( name, step);
```

3. Как переустановить значение генератора?

Для того чтобы установить генератор в другое значение, необходимо использовать команду SET GENERATOR:

```
SET GENERATOR name TO value;
```

4. Как удалить генератор?

```
DELETE FROM RDB$GENERATORS
```

```
WHERE RDB$GENERATORS_NAME = 'Имя_генератора';
```

5. Повышение надежности данных?

ДА, это важно. Триггер в основном используется для поддержания целостности информации в базе данных.

6. Организация многопользовательского режима доступа к данным?

Такое реализовать можно, но мы такое не изучали. А если и изучали, то я не знал.

7. Что такое «триггер»?

Триггер – процедура, автоматически вызываемая при операциях с таблицей.

8. Из каких частей состоит триггер?

Оператор состоит из заголовка триггера и тела триггера.

9. Какая информация содержится в заголовочной части триггера?

- имя триггера, уникальное по БД;
- имя таблицы, с которой ассоциируется триггер;
- указание на момент, когда триггер должен вызываться.

10. Как сделать триггер временно неактивным? Как удалить триггер?

ACTIVE|INACTIVE

Указывает «активность» триггера. Не активные триггеры игнорируются. По умолчанию триггер активен.

Удаление:

DROP TRIGGER <имя триггера>

11. Для чего используются триггеры?

Триггер в основном используется для поддержания целостности информации в базе данных.

12. Назовите элементы языка хранимых процедур и триггеров.

- BEGIN...END

определяет блок операторов (операторные скобки). Скобка после END не нужна.

- variable = expression

оператор присваивания

- /* comment_text */

многострочный комментарий

- EXCEPTION exception_name

вызывает исключительную ситуацию с именем exception_name, если она не обрабатывается оператором WHEN

- EXECUTE PROCEDURE

proc_name [var [, var ...]]

[RETURNING_VALUES var [, var ...]]

Вызывает хранимую процедуру с именем proc_name, с указанными входными и выходными параметрами. Параметры процедуры должны быть локальными переменными.

- FOR select_statement DO

compound_statement

повторяет выполнение блока кода следующего за DO для каждой строки, возвращенной оператором select_statement.

- select_statement.

select_statement - обычный запрос на выборку, за исключением того, что он обязательно должен содержать часть INTO, и данная часть должна идти на последнем месте.

- compound_statement

или одиночный оператор языка, или блок операторов заключенных в операторные скобки.

- IF (condition)

THEN compound_statement [ELSE

compound_statement]

условный оператор condition - условие, выражение булевской трехзначной логики (TRUE, FALSE, UNKNOWN). Обычно два выражения как операнды оператора сравнения.

- NEW.column

контекстная переменная, содержащая новое значение столбца с именем column при выполнении операций INSERT или UPDATE.

- OLD.column

контекстная переменная, содержащая старое значение столбца с именем column при выполнении операций INSERT или UPDATE.

- POST_EVENT event_name

отсылает сообщение с именем event_name.

- WHILE (condition) DO compound_statement

цикл с предусловием.

- WHEN {error [, error .]|ANY}

DO compound_statement

Оператор обработки исключительных ситуаций и ошибок. В случае если имеет место одна из ошибок перечисленных в операторе WHEN, вызывается compound_statement. Оператор WHEN должен быть последним оператором перед END тела триггера. Error: EXCEPTION exception_name, SQLCODE errcode or GDSCODE number. Ошибкой может быть - возбуждение исключительной ситуации с именем exception_name, равенство константы SQLCODE значению errcode, равенство

константы GDSCODE значению number. ANY – обрабатывает любую ошибку, если она имела место.