ЛАБОРАТОРНАЯ РАБОТ 4.
ОПЕРАТОРО JOIN. СОЕДИНЕНИЯ.

JOIN - оператор языка SQL, который является реализацией операции соединения реляционной алгебры. Входит в раздел FROM операторов SELECT, UPDATE или DELETE.

Операция соединения предназначена для обеспечения выборки данных из двух таблиц и включения этих данных в один результирующий набор. Отличительной особенностью операции соединения является следующее: в схему таблицы-результата входят столбцы обеих исходных таблиц (таблиц-операндов), то есть схема результата является

"сцеплением" схем операндов; каждая строка таблицы-результата является "сцеплением" строки из одной таблицы-операнда со строкой второй таблицы-операнда. Определение того, какие именно исходные строки войдут в результат и в каких сочетаниях, зависит от типа операции соединения и от явно заданного условия соединения. Условие соединения, то есть условие сопоставления строк исходных таблиц друг с другом, представляет собой логическое выражение (предикат).

При необходимости соединения не двух, а нескольких таблиц, операция соединения применяется несколько раз (последовательно)

Типы соединений

Существуют следующие типы соединений таблиц:
INNER JOIN
FULL OUTER JOIN
LEFT OUTER JOIN
RIGHT OUTER JOIN
CROSS JOIN

Для выполнения лабораторной работы рассмотрим пример базы данных, которая содержит следующие таблиц:

```
CREATE DATABASE TEST_JOIN ON PRIMARY
(
     NAME = 'TEST_JOIN',
     FILENAME = 'D:\TEST_JOIN.mdf' ,
     SIZE = 3072KB ,
     MAXSIZE = UNLIMITED,
     FILEGROWTH = 1024KB )
LOG ON
(
     NAME = 'TEST_JOIN_log',
     FILENAME = 'D:\TEST_JOIN_log.ldf' ,
     SIZE = 1024KB ,
     MAXSIZE = 2048GB ,
     FILEGROWTH = 10%
)
GO

CREATE TABLE PERSON
(
     ID INTEGER IDENTITY(1,1),
     NAME VARCHAR(50) NOT NULL,
     CID INTEGER NOT NULL,
     PRIMARY KEY (ID),
);
GO

CREATE TABLE CITY
(
     CID INTEGER IDENTITY(1,1),
     CITYNAME VARCHAR(50) NOT NULL,
     PRIMARY KEY (CID),
);
GO
```

Заполним таблицы данными

```
INSERT INTO PERSON VALUES
     ('Андрей', 1),
     ('Леонид', 2),
     ('Сергей', 1),
     ('Григорий', 4)

INSERT INTO CITY VALUES
     ('Москва'),
     ('Санкт-Петербург'),
     ('Казань')
GO*/
```

Оператор внутреннего соединения INNER JOIN

Оператор внутреннего соединения INNER JOIN соединяет две таблицы. Порядок таблиц для оператора неважен, поскольку оператор является симметричным. Заголовок таблицы-результата является объединением заголовков соединяемых таблиц. Тело результата логически формируется

следующим образом. Каждая строка одной таблицы сопоставляется с каждой строкой второй таблицы, после чего для полученной "соединённой" строки проверяется условие соединения (вычисляется предикат соединения). Если условие истинно, в таблицу-результат добавляется соответствующая "соединённая" строка.

Пример 1

```
--SELECT * FROM PERSON INNER JOIN CITY ON PERSON.CID = CITY.CID
--SELECT NAME, CITYNAME FROM PERSON INNER JOIN CITY ON PERSON.CID =
CITY.CID
--SELECT NAME, CITYNAME FROM PERSON INNER JOIN CITY ON PERSON.CID =
CITY.CID WHERE CITYNAME LIKE 'M%'
```

Оператор внутреннего соединения OUTER JOIN

Соединение двух таблиц, в результат которого в обязательном порядке входят строки либо одной, либо обеих таблиц.

Оператор полного внешнего соединения FULL OUTER JOIN соединяет две таблицы. Порядок таблиц для оператора неважен, поскольку оператор является симметричным. Заголовок таблицы-результата является объединением заголовков соединяемых таблиц. Тело результата логически формируется следующим образом. Пусть выполняется соединение первой и второй таблиц по предикату (условию) p. Слова «первой» и «второй» здесь не обозначают порядок в записи (который неважен), а используются лишь для различения таблиц. В результат включается внутреннее соединение (INNER JOIN) первой и второй таблиц по предикату p. В результат добавляются те записи первой таблицы, которые не вошли во внутреннее соединение на первом шаге. Для таких записей поля, соответствующие второй таблице, заполняются значениями NULL. В результат добавляются те записи второй таблицы, которые не вошли во внутреннее соединение на первом шаге. Для таких записей поля, соответствующие первой таблице, заполняются значениями NULL.

```
--SELECT * FROM PERSON FULL OUTER JOIN CITY ON PERSON.CID = CITY.CID
```

Оператор внутреннего соединения LEFT OUTER JOIN

Оператор левого внешнего соединения LEFT OUTER JOIN соединяет две таблицы. Порядок таблиц для оператора важен, поскольку оператор не является симметричным. Заголовок таблицы-результата является объединением заголовков соединяемых таблиц. Тело результата логически формируется следующим образом. Пусть выполняется соединение левой и

правой таблиц по предикату (условию) p. В результат включается внутреннее соединение (INNER JOIN) левой и правой таблиц по предикату p.

Затем в результат добавляются те записи левой таблицы, которые не вошли во внутреннее соединение на шаге. Для таких записей поля, соответствующие правой таблице, заполняются значениями NULL.

```
--SELECT * FROM PERSON LEFT OUTER JOIN CITY ON PERSON.CID = CITY.CID
```

Оператор внутреннего соединения RIGHT OUTER JOIN

Оператор правого внешнего соединения RIGHT OUTER JOIN соединяет две таблицы. Порядок таблиц для оператора важен, поскольку оператор не является симметричным. Заголовок таблицы-результата является объединением (конкатенацией) заголовков соединяемых таблиц. Тело результата логически формируется следующим образом. Пусть выполняется соединение левой и правой таблиц по предикату (условию) p. В результат включается внутреннее соединение (INNER JOIN) левой и правой таблиц по предикату p. Затем в результат добавляются те записи правой таблицы, которые не вошли во внутреннее соединение на шаге. Для таких записей поля, соответствующие левой таблице, заполняются значениями NULL.

```
--SELECT * FROM PERSON RIGHT OUTER JOIN CITY ON PERSON.CID = CITY.CID
```

Оператор внутреннего соединения CROSS JOIN

Оператор перекрёстного соединения, или декартова произведения CROSS JOIN соединяет две таблицы. Порядок таблиц для оператора неважен, поскольку оператор является симметричным. Заголовок таблицы-результата является объединением (конкатенацией) заголовков соединяемых таблиц. Тело результата логически формируется следующим образом. Каждая строка одной таблицы соединяется с каждой строкой второй таблицы, давая тем самым в результате все возможные сочетания строк двух таблиц.

```
--SELECT * FROM PERSON CROSS JOIN CITY
```