Представления.

- Представление VIEW – это именованная виртуальная таблица, содержание которой выбирается из других таблиц с помощью запросов.
- При изменении значений в таблицах автоматически меняются значения представления.
- Наличие имени у такой таблицы позволяет пользователю выполнять с ней операции аналогичные операциям с базовыми таблицами.

- Рассмотрим таблицы, относящиеся к базовым, т.е. таким, которые содержат данные и постоянно находятся на устройствах хранения информации. Представления по сравнению с ними являются более гибкими средствами. Когда СУБД отыскивает в команде ссылку на представление, она отыскивает его определение, хранящееся в БД.
- После этого происходит преобразование пользовательской команды в её эквивалент с учетом запроса. У пользователя возникает впечатление, что он работает с настоящей реально существующей таблицей.

- СУБД имеет две возможности реализации представления:
- - если определение представления простое, то система формирует каждую запись по мере необходимости;
- - если представление сложное, СУБД сначала выполняет материализацию представления, т.е. сохраняет информацию, из которой состоит представление во временной таблице. Затем система выполняет пользовательские команды и формирует её результаты, после временная таблица удаляется.

CREATE VIEW <Name of view > [(name of attributes),…] AS <SELECT …>;

- Пример 12.1. Создать представление о студентах, получающих стипендию в размере 25.50.
- **CREATE VIEW STIP25_50**

AS SELECT * FROM STUDENTS

WHERE STIP=25.50;

- Теперь в БД существует представление STIP25_50. Это такая же таблица, как и остальные. С ней можно выполнять запросы, изменения, вставки как с другими таблицами. При выполнении запроса к ней:

SELECT * FROM STIP25_50;

будет получена таблица.

SNUM	SFAM	SNAME	SFATH	STIP
3412	Поляков	Анатолий	Алексеевич	25,50
3416	Нагорный	Евгений	Васильевич	25,50

Если к ней обратиться с запросом с предикатом

SELECT * FROM STIP25_50 WHERE SFAM<’П’;

То будет получен результат:

SNUM	SFAM	SNAME	SFATH	STIP
3416	Нагорный	Евгений	Васильевич	25,50

При создании представлений можно часть информации скрыть.

- Пример 12.2. Создать представление о студентах без указания стипендии.

CREATE VIEW STIPOFF AS SELECT **SNUM, SFAM, SNAME, SFATH**

FROM STUDENTS;

- При выполнении запроса к ней, будет получена таблица :

SNUM	SFAM	SNAME	SFATH
3412	Поляков	Анатолий	Алексеевич
3413	Старова	Любовь	Михайловна
3414	Гриценко	Владимир	Николаевич
3415	Котенко	Анатолий	Николаевич
3416	Нагорный	Евгений	Васильевич

Представление теперь может изменяться также как и таблица, фактически же команда направлена к таблице STUDENTS.

- **Пример 12.3. Изменить у студента с номером студенческого билета 3415 имя.**
- **UPDATE STIPOFF SET SNAME =’ Василий’ WHERE SNUM=3415;**
- **UPDATE STUDENTS SET SNAME =’ Василий’ WHERE SNUM=3415;**

Но команда

- UPDATE STIPOFF SET STIP =100 WHERE SNUM=3415;

будет отвергнута, так как поле STIP в представлении STIPOFF отсутствует.

Существуют ограничения на модификацию представлений.

- Чаще всего в представлении используются те же имена столбцов, что и в базовых таблицах. При использовании объединения может быть конфликт имен. Допустимо указание других имен в представлении в скобках после имени.
- Пример 12.4.

```
CREATE VIEW STIPCOUNT(STIP, NUM) AS
SELECT STIP, COUNT(*) FROM STUDENTS
GROUP BY STIP;
```




- Сделаем запрос к представлению: показать все данные о стипендиях, которые получают 2 и более человек.

SELECT * FROM STIPCOUNT WHERE NUM>=2;

- Но не допускается функция в предикате WHERE:

SELECT STIP, COUNT(*) FROM STUDENTS WHERE COUNT(*) >= 2 GROUP BY STIP; Правильная команда:

SELECT STIP, COUNT(*) as NUM FROM STUDENTS GROUP BY STIP HAVING COUNT(*)>=2;

STIP	NUM
0,00	2
25,50	2

В SQL существует понятие групповых представлений, т.е. имеющих предложение GROUP BY или основанных на других групповых представлениях.

```
CREATE VIEW STIPCOUNT (STIP, NUM)
AS SELECT STIP, COUNT(*) FROM
STUDENTS GROUP BY STIP;
```

STIP	NUM
0,00	2
17,00	1
25,50	2

Представления могут основываться на двух и более таблицах.

- Пример 12.5.Создать представление о студентах и их оценках.
- CREATE VIEW STUDMARK AS SELECT C.UNUM, A.SFAM, B.PNAME, C.MARK

FROM STUDENTS A, PREDMET B, USP C WHERE A.SNUM=C.SNUM AND B.PNUM=C.PNUM;

- После этого легче ориентироваться в оценках:
- SELECT * FROM STUDMARK;

UNUM	SFAM	PNAME	MARK
1001	Поляков	Физика	5
1002	Старова	Математика	4
1003	Гриценко	Экономика	3
1004	Поляков	Математика	4
1005	Нагорный	Философия	5
1006	Гриценко	Физика	2
1007	Поляков	Экономика	4

- Допускается соединение представления с базовыми таблицами:

```
SELECT SFAM, PNAME, MARK, UDATE
FROM STUDMARK A, USP B
WHERE A.SFAM= ‘Поляков’
AND A.UNUM=B.UNUM;
```

Результат работы запроса:

SFAM	PNAME	MARK	UDATE
Поляков	Физика	5	10.06.2002
Поляков	Математика	4	12.06.2002
Поляков	Экономика	4	13.06.2002

Представления допускают соотнесенные подзапросы. Пример12.6.Пусть в таблице USP

UNUM	SNUM	PNUM	UDATE	MARK
1001	3412	2001	10.06.2002	5
1002	3413	2003	10.06.2002	4
1003	3414	2005	11.06.2002	3
1004	3412	2003	12.06.2002	4
1005	3416	2004	12.06.2002	5
1006	3414	2001	12.06.2002	2
1007	3412	2005	13.06.2002	4

- Создадим представление об оценках превышающих среднюю.
- 1)CREATE VIEW AVGMARK AS SELECT * FROM USP A WHERE MARK>(SELECT AVG(MARK) FROM USP B WHERE B.PNUM= A.PNUM);
- 2)SELECT * FROM AVGMARK;

UNUM	SNUM	PNUM	UDATE	MARK
1001	3412	2001	10.06.2002	5
1007	3412	2005	13.06.2002	4

Из этих примеров следует, что представления значительно облегчают работу с данными. Однако они являются чаще всего объектами доступными для чтения.

- Существуют ограничения на построения представлений:

1) в них не допускаются объединения UNION запросов;
2) не допустимо упорядочение ORDER BY, так как в базовых таблицах записи не упорядочены.

Для удаления представлений используется команда DROP VIEW <Name of view>;

- Для удаления представления не требуется удалять все данные, потому что реально они в нем не содержатся.
- DROP VIEW AVGMARK;

- Так как представления состоят из результатов запросов, то для их модификации должны быть модифицированы данные из базовых таблиц. Но модификация не должна воздействовать на запрос, она воздействует на значения в таблице.

- Рассмотрим критерии, по которым мы определяем, является ли представление модифицируемым:

1) представление должно основываться только на одной таблице;
2) оно должно содержать первичный ключ этой таблицы;
3) представление не должно иметь полей - агрегатных функций;

4) представление не должно использовать DISTINCT;

5) представление не должно использовать GROUP BY, HAVING;

6) представление не должно использовать подзапросы;

7) представление не должно использовать константы, строки, выражения среди полей вывода;

8) для команды INSERT оно может содержать любые поля базовой таблицы, для которой имеются ограничения NOT NULL, если другое значение по умолчанию не определено.

- Модификация представлений подобна фрагментации базовых таблиц.
- Пример 12.7.

CREATE VIEW PRCOUNT (UDATE, COL) AS SELECT UDATE, COUNT(*) FROM USP GROUP BY UDATE;

Это представление не модифицируемо -

GROUP BY.

- Пример 12.8.

```
CREATE VIEW   MATEMUSP
 AS SELECT * FROM USP
WHERE PNUM = 2003;
```

Это представление – модифицируемо.

- Другой результат достигается на представлении:

1)CREATE VIEW ONLY5
AS SELECT SNUM, MARK
FROM USP
WHERE MARK = 5;
2)INSERT INTO ONLY5
VALUES (3415, 4);

Это допустимая команда, в таблицу эти значения будут вставлены, но на экране не появятся.

- Таким образом, в таблице могут появляться данные не видимые пользователю.
- Для исключения таких моментов используется предложение:

WITH CHECK OPTION.

Если его добавить к команде:

```
CREATE VIEW   ONLY5
AS SELECT SNUM, MARK
FROM USP
WHERE MARK = 5
      WITH CHECK OPTION;
```

То любое значение отличное от указанных будет отключено.

- Различия между модифицируемым представлением и представлением только для чтения существуют:
- Первое - работает как базовое, является в основном средством скрытия части информации, средством защиты;
- Второе - позволяет получать целый набор всевозможных запросов, которые можно повторять и использовать для других запросов.

- CREATE VIEW DATEMARK

AS SELECT SNUM, SFAM FROM STUDENTS WHERE SNUM IN (SELECT SNUM FROM USP WHERE UDATE = 10.06.2002);

Это представление для чтения – имеется подзапрос.

- Что будет если пользователь решит добавить запись:

INSERT INTO DATEMARK VALUES (3415,’ Котенко’);

Часть данных будет заполнена как NULL.

Проблема не решится если применить

WITH CHECK OPTION, так как представление станет модифицируемым и удаляемым, но без вставки.

- Пример 12.9.

1) CREATE VIEW STIPSTUD
   AS SELECT SNUM, SFAM, STIP
FROM STUDENTS WHERE STIP>0
WITH CHECK OPTION;

2) То вставка будет не удачна.

INSERT INTO STIPSTUD
VALUES (3417, Решетник, 0.00);

3)однако в NEW1 она возможна.

CREATE VIEW NEW1
   AS SELECT * FROM STIPSTUD;

- Вставка выполнится.Это означает, что любое корректное представление модифицируемо. Даже если:

```
CREATE VIEW NEW1
   AS SELECT * FROM STIPSTUD
WITH CHECK OPTION;
```

Стандарт	Input/Output**
Default	mon dd yyyy hh:miAM (or PM)
USA	mm/dd/yy
ANSI	yy.mm.dd
German/British /French/Italian	dd/mm/yy dd-mm-yy

Использование CAST:

CAST **(** *expression* AS *data_type* **)**

Использование CONVERT:

CONVERT **(** *data_type* [**(** *length* **)**] **,** *expression* [**,** *style*] **)**

```
-- Use CAST.
USE pubs
GO
SELECT SUBSTRING(title, 1, 30) AS Title, ytd_sales
FROM titles WHERE CAST(ytd_sales AS char(20)) LIKE '3%'
GO

 -- Use CONVERT.
USE pubs
GO
SELECT SUBSTRING(title, 1, 30) AS Title, ytd_sales
FROM titles WHERE CONVERT(char(20), ytd_sales) LIKE
'3%'
GO
```

Представления.

- Представление VIEW – это именованная виртуальная таблица, содержание которой выбирается из других таблиц с помощью запросов.
- При изменении значений в таблицах автоматически меняются значения представления.
- Наличие имени у такой таблицы позволяет пользователю выполнять с ней операции аналогичные операциям с базовыми таблицами.

- Рассмотрим таблицы, относящиеся к базовым, т.е. таким, которые содержат данные и постоянно находятся на устройствах хранения информации. Представления по сравнению с ними являются более гибкими средствами. Когда СУБД отыскивает в команде ссылку на представление, она отыскивает его определение, хранящееся в БД.
- После этого происходит преобразование пользовательской команды в её эквивалент с учетом запроса. У пользователя возникает впечатление, что он работает с настоящей реально существующей таблицей.

- СУБД имеет две возможности реализации представления:
- - если определение представления простое, то система формирует каждую запись по мере необходимости;
- - если представление сложное, СУБД сначала выполняет материализацию представления, т.е. сохраняет информацию, из которой состоит представление во временной таблице. Затем система выполняет пользовательские команды и формирует её результаты, после временная таблица удаляется.

CREATE VIEW <Name of view > [(name of attributes),…] AS <SELECT …>;

- Пример 12.1. Создать представление о студентах, получающих стипендию в размере 25.50.
- **CREATE VIEW STIP25_50**

AS SELECT * FROM STUDENTS

WHERE STIP=25.50;

- Теперь в БД существует представление STIP25_50. Это такая же таблица, как и остальные. С ней можно выполнять запросы, изменения, вставки как с другими таблицами. При выполнении запроса к ней:

SELECT * FROM STIP25_50;

будет получена таблица.

SNUM	SFAM	SNAME	SFATH	STIP
3412	Поляков	Анатолий	Алексеевич	25,50
3416	Нагорный	Евгений	Васильевич	25,50

Если к ней обратиться с запросом с предикатом

SELECT * FROM STIP25_50 WHERE SFAM<’П’;

То будет получен результат:

SNUM	SFAM	SNAME	SFATH	STIP
3416	Нагорный	Евгений	Васильевич	25,50

При создании представлений можно часть информации скрыть.

- Пример 12.2. Создать представление о студентах без указания стипендии.

CREATE VIEW STIPOFF AS SELECT **SNUM, SFAM, SNAME, SFATH**

FROM STUDENTS;

- При выполнении запроса к ней, будет получена таблица :

SNUM	SFAM	SNAME	SFATH
3412	Поляков	Анатолий	Алексеевич
3413	Старова	Любовь	Михайловна
3414	Гриценко	Владимир	Николаевич
3415	Котенко	Анатолий	Николаевич
3416	Нагорный	Евгений	Васильевич

Представление теперь может изменяться также как и таблица, фактически же команда направлена к таблице STUDENTS.

- **Пример 12.3. Изменить у студента с номером студенческого билета 3415 имя.**
- **UPDATE STIPOFF SET SNAME =’ Василий’ WHERE SNUM=3415;**
- **UPDATE STUDENTS SET SNAME =’ Василий’ WHERE SNUM=3415;**

Но команда

- UPDATE STIPOFF SET STIP =100 WHERE SNUM=3415;

будет отвергнута, так как поле STIP в представлении STIPOFF отсутствует.

Существуют ограничения на модификацию представлений.

- Чаще всего в представлении используются те же имена столбцов, что и в базовых таблицах. При использовании объединения может быть конфликт имен. Допустимо указание других имен в представлении в скобках после имени.
- Пример 12.4.

```
CREATE VIEW STIPCOUNT(STIP, NUM) AS
SELECT STIP, COUNT(*) FROM STUDENTS
GROUP BY STIP;
```




- Сделаем запрос к представлению: показать все данные о стипендиях, которые получают 2 и более человек.

SELECT * FROM STIPCOUNT WHERE NUM>=2;

- Но не допускается функция в предикате WHERE:

SELECT STIP, COUNT(*) FROM STUDENTS WHERE COUNT(*) >= 2 GROUP BY STIP; Правильная команда:

SELECT STIP, COUNT(*) as NUM FROM STUDENTS GROUP BY STIP HAVING COUNT(*)>=2;

STIP	NUM
0,00	2
25,50	2

В SQL существует понятие групповых представлений, т.е. имеющих предложение GROUP BY или основанных на других групповых представлениях.

```
CREATE VIEW STIPCOUNT (STIP, NUM)
AS SELECT STIP, COUNT(*) FROM
STUDENTS GROUP BY STIP;
```

STIP	NUM
0,00	2
17,00	1
25,50	2

Представления могут основываться на двух и более таблицах.

- Пример 12.5.Создать представление о студентах и их оценках.
- CREATE VIEW STUDMARK AS SELECT C.UNUM, A.SFAM, B.PNAME, C.MARK

FROM STUDENTS A, PREDMET B, USP C WHERE A.SNUM=C.SNUM AND B.PNUM=C.PNUM;

- После этого легче ориентироваться в оценках:
- SELECT * FROM STUDMARK;

UNUM	SFAM	PNAME	MARK
1001	Поляков	Физика	5
1002	Старова	Математика	4
1003	Гриценко	Экономика	3
1004	Поляков	Математика	4
1005	Нагорный	Философия	5
1006	Гриценко	Физика	2
1007	Поляков	Экономика	4

- Допускается соединение представления с базовыми таблицами:

```
SELECT SFAM, PNAME, MARK, UDATE
FROM STUDMARK A, USP B
WHERE A.SFAM= ‘Поляков’
AND A.UNUM=B.UNUM;
```

Результат работы запроса:

SFAM	PNAME	MARK	UDATE
Поляков	Физика	5	10.06.2002
Поляков	Математика	4	12.06.2002
Поляков	Экономика	4	13.06.2002

Представления допускают соотнесенные подзапросы. Пример12.6.Пусть в таблице USP

UNUM	SNUM	PNUM	UDATE	MARK
1001	3412	2001	10.06.2002	5
1002	3413	2003	10.06.2002	4
1003	3414	2005	11.06.2002	3
1004	3412	2003	12.06.2002	4
1005	3416	2004	12.06.2002	5
1006	3414	2001	12.06.2002	2
1007	3412	2005	13.06.2002	4

- Создадим представление об оценках превышающих среднюю.
- 1)CREATE VIEW AVGMARK AS SELECT * FROM USP A WHERE MARK>(SELECT AVG(MARK) FROM USP B WHERE B.PNUM= A.PNUM);
- 2)SELECT * FROM AVGMARK;

UNUM	SNUM	PNUM	UDATE	MARK
1001	3412	2001	10.06.2002	5
1007	3412	2005	13.06.2002	4

Из этих примеров следует, что представления значительно облегчают работу с данными. Однако они являются чаще всего объектами доступными для чтения.

- Существуют ограничения на построения представлений:

1) в них не допускаются объединения UNION запросов;
2) не допустимо упорядочение ORDER BY, так как в базовых таблицах записи не упорядочены.

Для удаления представлений используется команда DROP VIEW <Name of view>;

- Для удаления представления не требуется удалять все данные, потому что реально они в нем не содержатся.
- DROP VIEW AVGMARK;

- Так как представления состоят из результатов запросов, то для их модификации должны быть модифицированы данные из базовых таблиц. Но модификация не должна воздействовать на запрос, она воздействует на значения в таблице.

- Рассмотрим критерии, по которым мы определяем, является ли представление модифицируемым:

1) представление должно основываться только на одной таблице;
2) оно должно содержать первичный ключ этой таблицы;
3) представление не должно иметь полей - агрегатных функций;

4) представление не должно использовать DISTINCT;

5) представление не должно использовать GROUP BY, HAVING;

6) представление не должно использовать подзапросы;

7) представление не должно использовать константы, строки, выражения среди полей вывода;

8) для команды INSERT оно может содержать любые поля базовой таблицы, для которой имеются ограничения NOT NULL, если другое значение по умолчанию не определено.

- Модификация представлений подобна фрагментации базовых таблиц.
- Пример 12.7.

CREATE VIEW PRCOUNT (UDATE, COL) AS SELECT UDATE, COUNT(*) FROM USP GROUP BY UDATE;

Это представление не модифицируемо -

GROUP BY.

- Пример 12.8.

```
CREATE VIEW   MATEMUSP
 AS SELECT * FROM USP
WHERE PNUM = 2003;
```

Это представление – модифицируемо.

- Другой результат достигается на представлении:

1)CREATE VIEW ONLY5
AS SELECT SNUM, MARK
FROM USP
WHERE MARK = 5;
2)INSERT INTO ONLY5
VALUES (3415, 4);

Это допустимая команда, в таблицу эти значения будут вставлены, но на экране не появятся.

- Таким образом, в таблице могут появляться данные не видимые пользователю.
- Для исключения таких моментов используется предложение:

WITH CHECK OPTION.

Если его добавить к команде:

```
CREATE VIEW   ONLY5
AS SELECT SNUM, MARK
FROM USP
WHERE MARK = 5
      WITH CHECK OPTION;
```

То любое значение отличное от указанных будет исключено.

- Различия между модифицируемым представлением и представлением только для чтения существуют:
- Первое - работает как базовое, является в основном средством скрытия части информации, средством защиты;
- Второе - позволяет получать целый набор всевозможных запросов, которые можно повторять и использовать для других запросов.

- CREATE VIEW DATEMARK

AS SELECT SNUM, SFAM FROM STUDENTS WHERE SNUM IN (SELECT SNUM FROM USP WHERE UDATE = 10.06.2002);

Это представление для чтения – имеется подзапрос.

- Что будет если пользователь решит добавить запись:

INSERT INTO DATEMARK VALUES (3415,’ Котенко’);

Часть данных будет заполнена как NULL.

Проблема не решится если применить

WITH CHECK OPTION, так как представление станет модифицируемым и удаляемым, но без вставки.

- Пример 12.9.

1) CREATE VIEW STIPSTUD
   AS SELECT SNUM, SFAM, STIP
FROM STUDENTS WHERE STIP>0
WITH CHECK OPTION;

2) То вставка будет не удачна.

INSERT INTO STIPSTUD
VALUES (3417, Решетник, 0.00);

3)однако в NEW1 она возможна.

CREATE VIEW NEW1
   AS SELECT * FROM STIPSTUD;

Стандарт	Input/Output**
Default	mon dd yyyy hh:miAM (or PM)
USA	mm/dd/yy
ANSI	yy.mm.dd
German/British /French/Italian	dd/mm/yy dd-mm-yy

Использование CAST:

CAST **(** *expression* AS *data_type* **)**

Использование CONVERT:

CONVERT **(** *data_type* [**(** *length* **)**] **,** *expression* [**,** *style*] **)**

```
-- Use CAST.
USE pubs
GO
SELECT SUBSTRING(title, 1, 30) AS Title, ytd_sales
FROM titles WHERE CAST(ytd_sales AS char(20)) LIKE '3%'
GO

 -- Use CONVERT.
USE pubs
GO
SELECT SUBSTRING(title, 1, 30) AS Title, ytd_sales
FROM titles WHERE CONVERT(char(20), ytd_sales) LIKE
'3%'
GO
```

Временные таблицы

Временная таблица создается командой create table, также как и обычная таблица, признаком временности служит символ # перед именем (такая таблица доступна только в текущей подпрограмме, таблица с префиксом ## всюду). Временные таблицы автоматически уничтожаются при завершении текущей сессии работы с сервером, в остальном ничем не отличаются от обычных таблиц.