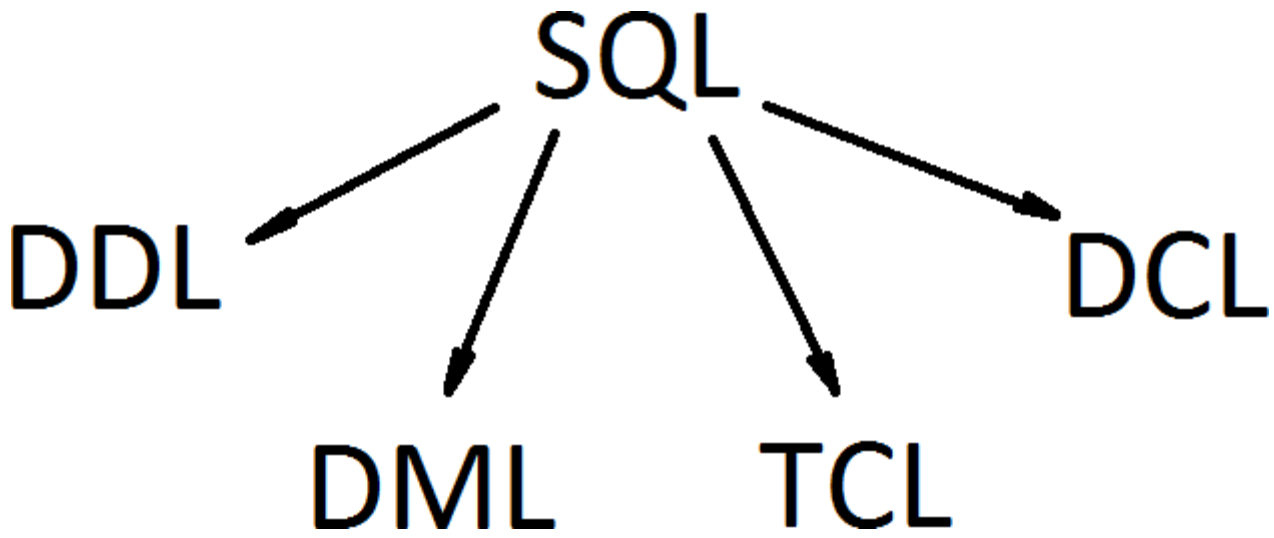
База Данных

Лекция 7

Реляционная алгебра

Структура SQL



- DDL (Язык определения данных)
- DML (Язык манипулирования данными)
- TCL (Язык управления транзакциями)
- DCL (Язык управления данными)

Прошлая лекция

DML - это язык, который позволяет получать доступ к данным и манипулировать ими.

Операторы DML:

- INSERT
- UPDATE
- DELETE
- SELECT

Извлечение данных из Таблиц

- SQL позволяет извлекать данные с помощью инструкции SELECT.
- Синтаксис:

  SELECT attribute(s) FROM table(s)
  [WHERE selection condition(s)] ;

- Выборка всех данных из таблицы:

  SELECT * FROM название_таблицы;
- Выборка определенных данных из таблицы:

  SELECT столбец1, столбец2 FROM название_таблицы WHERE условие;

Реляционная алгебра

- Реляционная алгебра разработана в 1970-х годах Эдгаром Коддом, основателем реляционных баз данных. Основная **цель реляционной алгебры** - обеспечить эффективное извлечение данных из базы данных путем выполнения различных операций над отношениями (таблицами).

- **Реляционная алгебра (РА)** – это теоретический язык операций, который на основе одного или нескольких отношений (таблиц) позволяет создавать другое отношение (таблицу) без изменения исходных данных.

- Операции РА применяются к отношениям и в результате применения операций РА получаются отношения (таблицы). Таким образом, оба операнда и результат являются отношениями.

Реляционная алгебра

- Аналогично обычной алгебре, за исключением того, что мы используем отношения в качестве значений вместо чисел, а также операции и операторы другие.
- Основное применение реляционной алгебры - обеспечение теоретической основы для реляционных баз данных.
- Не используется в качестве языка запросов в реальных СУБД (вместо этого используется SQL).
- Нам нужно знать о реляционной алгебре, чтобы понять выполнение запросов в реляционной СУБД.

Реляционное свойство замкнутости

- **Реляционное свойство замкнутости** говорит о том, что результат любой операции над двумя отношениями (таблицами) в реляционной алгебре также является отношением.
- То есть, если у нас есть две таблицы с данными, и мы применяем к ним операцию (например, выборку, проекцию, объединение и т.д.), то результат этой операции будет также таблицей с данными, а значит, это тоже будет отношением.

Операции реляционной алгебры

- выборка (selection)
- проекция (projection)
- декартово произведение (cartesian product)
- объединение (union)
- разность (set difference)
- соединение (join)
- пересечение (intersection)
- деление (division)

Унарные и бинарные операции

Операции выборки и проекции являются ***унарными***, поскольку они работают с одним отношением.

Другие операции работают с парами отношений, и поэтому **их** называют ***бинарными*** операциями.

Выборка (selection)

$\sigma_{предикат}(R)$

Операция выборки работает с одним отношением R и определяет результирующее отношение, которое содержит только те кортежи (строки) отношения R, которые удовлетворяют заданному условию (предикату).

- Выборка (selection) обозначается как σ (сигма)
- Выбирает набор строк из таблицы, удовлетворяющих условию выбора
- Условием выбора является индекс σ, а таблица заключена в круглые скобки

$\sigma_{gpa>3}(Students)$

Выборка

Выборка (selection)

Выборка (selection) на языке SQL реализуется в разделе WHERE команды SELECT.

Например, выборка

$\sigma_{gpa>3}(Students)$

на SQL может быть реализована в следующем виде:

```
SELECT * FROM Students
WHERE gpa>3;
```

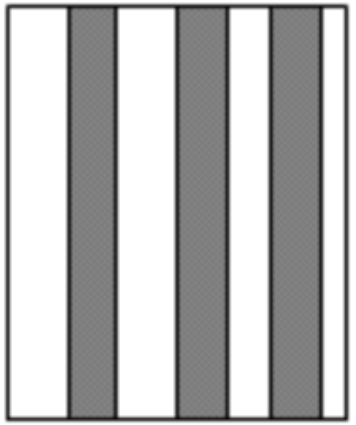
Проекция (projection)

$\Pi_{атр\ 1, \ldots, атр\ n}(R)$

Операция проекции работает с одним отношением R и определяет новое отношение, содержащее вертикальное подмножество отношения R, создаваемое посредством извлечения значений указанных атрибутов и исключения из результата строк-дубликатов.

- Проекция (projection) обозначается как П (pi)
- Выбирает набор атрибутов из таблицы
- Атрибуты являются индексами к П, а таблица заключена в круглые скобки

$\Pi_{stud\_id}(Students)$



Проекция

- Проекция реализована в списке **атрибутов** инструкции SQL SELECT. Приведенная выше проекция является синонимом следующего SQL-запроса:

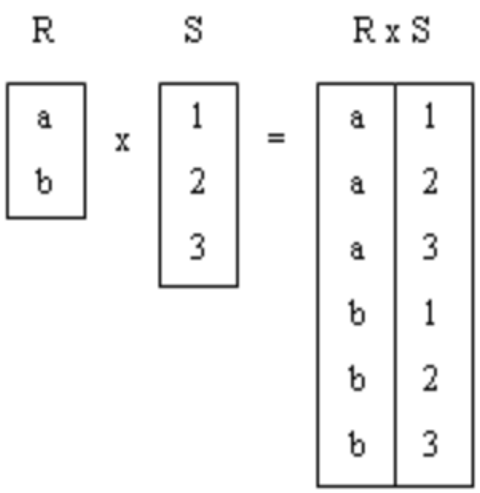
```
SELECT stud_id FROM Students;
```

Декартово произведение (cartesian product)

R × S

Операция декартового произведения определяет новое отношение, которое является результатом конкатенации (комбинирования) каждого кортежа (строки) отношения (таблицы) R с каждым кортежем (строкой) отношения (таблицы) S.

Декартово произведение (Cartesian product) в реляционной алгебре обозначается символом × (крестик) и представляет собой операцию, при которой каждая строка из одной таблицы комбинируется с каждой строкой из другой таблицы. Результатом является новая таблица, содержащая все возможные комбинации строк из исходных таблиц.



Декартово произведение

Декартово произведение (cartesian product)

Предположим, у нас есть две таблицы в базе данных университета: "Студенты" и "Курсы". Первая таблица содержит информацию о студентах, такую как их имена и ИИН, а вторая таблица содержит информацию о курсах, например, их названия и шифр курса.

Пример декартова произведения этих двух таблиц может выглядеть так:

- $(\Pi_{f\_name,\ IIN}(Students)) \times (\Pi_{course\_title,\ code}(Courses))$

Для реализации декартова произведения (cartesian product) таблиц в SQL вы можете использовать оператор JOIN без указания условия соединения между таблицами.

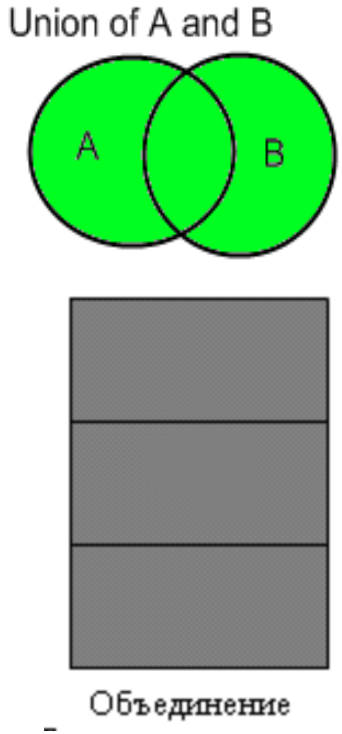
```
SELECT Students.f_name, Students.IIN, Courses. course_title, Courses.code FROM Students, Courses;
```

Объединение (union)

R ∪ S

Объединение двух отношений *R и S* определяет новое отношение, которое включает все кортежи, содержащиеся только в R, или только в *S или* одновременно в *R и S ,* причем все дубликаты кортежей исключены. При этом отношения R и *S* должны быть совместимыми по объединению.

Объединение отношений возможно только в том случае, когда они имеют одинаковое число атрибутов с совпадающими доменами (типами данных). Такие отношения являются **совместимыми по объединению**.



Совместимость по объединению

Две таблицы должны быть совместимы по объединению:

- таблицы должны иметь одинаковое количество атрибутов;
- тип данных каждого атрибута также должен быть одинаковым.

Table R		
A	**B**	**C**
1	2	3
4	5	6
7	8	9

Table S		
A	**B**	**C**
1	3	2
4	5	6
8	7	9

Объединение (union)

- Операция Объединения (union) для отношения A UNION отношения B, обозначается как **A∪B**, включает все кортежи, которые находятся в A или в B, исключая повторяющиеся кортежи.

- Для Объединения (union) SQL поддерживает оператор **UNION**.
- Синтаксис SQL:

```
SELECT * From R  UNION
SELECT * From S
```

Чтобы включить дубликаты, используйте оператор **UNION ALL.**

UNION

SELECT * From
R UNION
SELECT * From S

Table R		
A	B	C
1	2	3
4	5	6
7	8	9

Table S		
A	B	C
1	3	2
4	5	6
8	7	9

Table R U S		
A	B	C
1	2	3
4	5	6
7	8	9
1	3	2
8	7	9

UNION ALL

SELECT * From R

UNION ALL

SELECT * From S

Table R		
A	B	C
1	2	3
4	5	6
7	8	9

Table S		
A	B	C
1	3	2
4	5	6
8	7	9

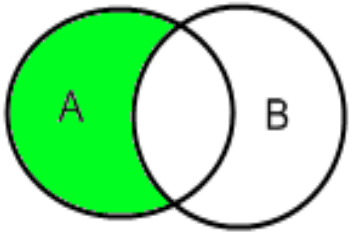
Table R U S		
A	B	C
1	2	3
4	5	6
7	8	9
1	3	2
4	5	6
8	7	9

Разность (set difference)

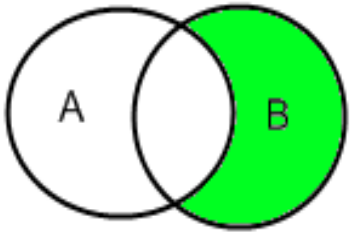
R - S

Разность двух отношений *R* и *S* состоит из кортежей, которые есть в отношении *R*, но отсутствуют в отношении *S*. Причем отношение *R* и *S* должны быть совместимыми по объединению.

Разность (set difference) (R - S) - это отношение, содержащее все кортежи R, которые не отображаются в S.



Difference A minus B



Difference B minus A


Разность (множеств)

Set Difference / EXCEPT

- Операция Разность (set difference) включает кортежи из одного отношения, которых нет в другом отношении.
- Пусть отношениями являются A и B, операция A ЗА ИСКЛЮЧЕНИЕМ B, обозначается A – B, что приводит к кортежам, которые принадлжат A, а не B.
- Для Разности (set difference) SQL поддерживает оператор **EXCEPT**.
- Синтаксис SQL:
- SELECT * FROM A EXCEPT
  SELECT * FROM B

EXCEPT

```
SELECT * FROM R
EXCEPT
SELECT * FROM S
```

Table R		
A	**B**	**C**
1	2	3
4	5	6
7	8	9

Table S		
A	**B**	**C**
1	3	2
4	5	6
8	7	9

Table R - S		
A	**B**	**C**
1	2	3
7	8	9

EXCEPT

```
SELECT * FROM S
EXCEPT
SELECT * FROM R
```

Table R		
A	**B**	**C**
1	2	3
4	5	6
7	8	9

Table S		
A	**B**	**C**
1	3	2
4	5	6
8	7	9

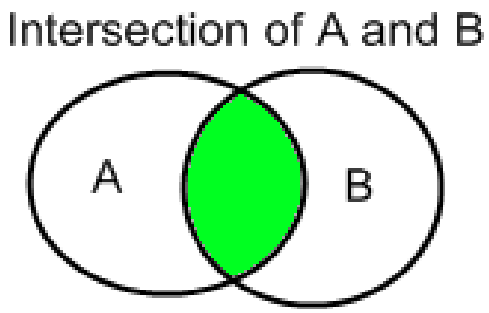
Table S - R		
A	**B**	**C**
1	3	2
8	7	9

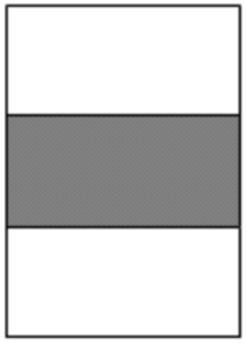
Пересечение (intersection)

R ∩ S

Операция пересечения определяет отношение, содержащее кортежи, которые есть, как в отношении R, так и в отношении S. Отношение R и S должны быть совместимы по объединению.

Операцию пересечения можно записать, используя операции разницы между отношениями: $R \cap S = R - (R - S)$




Пересечение

Пересечение (intersection)

Отношение A ПЕРЕСЕКАЕТ отношение B, обозначается ка **A ∩ B**, включает кортежи, которые находятся только в A и B.

Другими словами, в результат включаются только кортежи, принадлежащие A и B или совместно используемые как A, так и B.

Для Пересечения (intersection) SQL поддерживает оператор **INTERSECT**.

Синтаксис SQL:

```
SELECT * FROM A
INTERSECT
SELECT * FROM B
```

INTERSECT

```
SELECT * FROM R
INTERSECT
SELECT * FROM S
```

Table R		
A	**B**	**C**
1	2	3
4	5	6
7	8	9

Table S		
A	**B**	**C**
1	3	2
4	5	6
8	7	9

Table R ∩ S		
A	**B**	**C**
4	5	6

Объединение запросов

Результаты двух запросов могут быть объединены с помощью заданных операций объединения, пересечения и разности.
Синтаксис таков

```
query1 UNION [ALL] query2
query1 INTERSECT [ALL] query2
query1 EXCEPT [ALL] query2
```

query1 и query2 - это запросы, которые могут использовать любую из операций, рассмотренных до этого момента.

Объединение запросов

Операции также могут быть вложенными, например
query1 UNION ***query2*** UNION ***query3***

который выполняется следующим образом:

(***query1*** UNION ***query2***) UNION ***query3***

Чтобы вычислить объединение, пересечение или разность двух таблиц, эти две таблицы должны быть "совместимыми по объединению", то есть иметь таблицы должны иметь одинаковое количество атрибутов, тип данных каждого атрибута также должен быть одинаковым.

Книги

- Коннолли, Томас М. Системы баз данных: практический подход к проектированию, внедрению и управлению / Томас М. Коннолли, Кэролин Э. Бегг.- Соединенные Штаты Америки: Pearson Education

- Гарсия-Молина, Х. Система баз данных: Полная книга / Эктор Гарсия-Молина.- Соединенные Штаты Америки: Пирсон Прентис Холл

- Шарма, Н. Основы баз данных: Книга для сообщества от сообщества / Нирадж Шарма, Ливиу Перниу.- Канада

- www.postgresql.org/docs/manuals/