Лекція 8-2020_2021

- На попередній лекції
- SQL. Стандарти. Розділи
- Типи даних в SQL
- Синтаксис команди SELECT
- Послідовність виконання запитів на вибірку
- Розділ FROM, оператор JOIN

На прошлой лекции

Операция выборки − построение горизонтального подмножества.
Операция проекции − построение вертикального подмножества.
Декартово произведение R*S − конкатенация каждого кортежа из отношения R с каждым кортежем из отношения S.
Соединение по эквивалентности − предикат F содержит только оператор равенства (=).
Левое внешнее соединение R и S − все кортежи из R, а из S только то, что отвечает условию соединения.
Правое внешнее соединение R и S − все из S, а из R только то, что отвечает условию соединения.

На прошлой лекции

Полусоединение − кортежи только одного отношения из соединения двух отношений R и S.

Отношения совместимы по объединению, если они обладают одинаковыми заголовками.

Объединение − реализация за счет использования SQL оператора UNION.

Пересечение − реализация за счет использования условия в SQL операторе SELECT.

Взятие разности − реализация за счет использования в условии SQL оператора SELECT предиката NOT IN.

На прошлой лекции

Дополнительные операции:

• переименование – изменение имени атрибута (AS);

• расширение – добавление нового вычисляемого атрибута;

• вставка – добавление строк (INSERT);

• обновление – изменение значений (UPDATE);

• удаление – удаление строк (DELETE);

• подведение итогов – вертикальные или групповые вычисления:

SQL

В 1970 году Эдгар Кодд − реляционная модель данных. В IBM экспериментальный проект **SystemR,** чтобы доказать практичность реляционной модели. Результат проекта – разработка SQL (structured query language).

SQL – информационно-логический язык, предназначенным для описания, изменения и извлечения данных, хранимых в РБД. SQL не является тьюринг-полным языком программирования, но стандарт допускает процедурные расширения языка.

Язык программирования тьюринг-полный, если на нём можно реализовать любую вычислимую функцию.

SQL содержит только набор стандартных операторов доступа к данным, хранящимся в базе данных.

SQL

SQL является непроцедурным языком. Термин «непроцедурный» означает, что на этом языке можно сформулировать, что именно нужно сделать с данными, но нельзя проинструктировать, как это следует сделать. В языке отсутствуют алгоритмические конструкции, такие как метки, операторы цикла, условные переходы.

Использование SQL: интерактивная работа, в прикладных программах. В прикладных программах SQL используются как встроенный язык для обращения к базе данных.

SQL

Стандарты ANSI (American National Standards Institute). :

1986 − SQL87 − Первый вариант стандарта;

1989 − SQL89 − Немного доработанный вариант SQL87;

1992 − SQL92 − уровень Entry Level стандарта SQL-92

1999 – регулярные выражения, рекурсивные запросы, триггеры, процедурные расширения, нескалярные типы данных, некоторые ОО возможности;

2003 г. – XML-данные, оконные функции для OLAP БД.

2006 г., 2008 г – в запросах SQL и Xquery, улучшены оконные функции.

2011 – реализована поддержка хронологических БД.

2016 – защита на уровне строк, полиморфные табличные функции

SQL, преимущества

1. Независимость от конкретной СУБД, если не используются специфические диалекты СУБД.
2. Наличие стандартов способствует «стабилизации» языка. (Сами стандарты сложные, например, SQL:2003 – более чем 1300 страниц текста).
3. Декларативность. Программист описывает на SQL только то, какие данные нужно извлечь или модифицировать. Как это сделать, решает СУБД при обработке SQL-запроса. Профессионализм, когда разработчик представляет, как СУБД будет разбирать текст его запроса. Сложный запрос может допускать несколько вариантов написания, различных по скорости выполнения, но одинаковых по результатам.

SQL, недостатки

1. Не полное соответствие реляционной модели данных. Например, SQL поддерживает неопределённые значения (NULL) или использует порядок колонок, т.к. разрешает ссылки на колонки по номерам
2. Сложность. Задумывался как средство работы конечного пользователя, из-за сложности превратился в инструмент программиста.
3. Отступления от стандартов. Разработчики СУБД вносят изменения в язык SQL, отступая от стандарта. Таким образом появляются диалекты языка SQL для каждой конкретной СУБД.
4. Сложность работы с иерархическими структурами. Только в SQL:2003 стандартизована рекурсивная конструкция WITH из диалекта SQL DB2.

Разделы SQL

DDL (Data Definition Language) – язык определения данных.

Позволяет создавать и изменять структуру объектов БД

(**`Create Table, Alter Table, Drop Table, Create DataBase, Alter DataBase, Create Index, ..., Create View`**);

DML (Data Manipulation Language) – язык манипулирования данными. Позволяет изменять, удалять, вставлять данные в объекты БД (**`Insert, Update, Delete`**);

DQL (Data Query Language) – язык запросов. Используется для выборки данных (**`Select`**);

DCL (Data Control Language) – язык управления данными. Содержит команды, определяющие доступ к информации (**`Grant, Revoke`**).

Типы данных в SQL

Стандарт ANSI SQL распознает только текстовый и числовой типы, но все коммерческие СУБД используют и другие специальные типы данных.

Access:

Setting	Type of data	Size
Text	Любое сочетание символов и цифр	до 255 симв.
Memo	Любое сочетание символов и цифр	до 65 535 симв.
Number	Числовые данные	1, 2, 4, 8 байтов
DateTime	Дата и временя от 100 года до 9999	8 байтов
Currency	Денежное значение (15, 4)	8 байтов
Boolean	Логический тип (Yes/No, True/False)	1 байт

Типы данных в SQL

Setting	Type of data	Size
Counter	Уникальная последовательность с автоинкрементированием на 1	4 байта
OLE Object	Внедренный объект (Microsoft Excel, Microsoft Word док., график, звук, изображение и др.)	до 1 Гб
Hyperlink	Ссылка (адрес в UNC или URL формате. Формирование адреса ➔ щелчок на Hyperlink в Insert меню).	до 2048 симв.

URL ➔ 198. 165. 24. 1

UNC ➔ C:\ DataBases\Link

Типы данных в SQL

В MySQL есть те же типы данных, но есть и другие:

TIMESTAMP – Временн***а***я метка Столбик TIMESTAMP обновляется автоматически при вставке или модификации значений в таблице, вносятся значения даты и времени самой последней операции. Может быть лишь одно поле этого типа в таблице.

DECIMAL (M[,D]) – ``Неупакованное" число с плавающей точкой. Ведет себя подобно столбцу CHAR, содержащему цифровое значение.

TIME, YEAR [(2|4)]

BLOB – Двоичный объект большого размера, который может содержать переменное количество данных.

Типы данных в SQL

CHAR(M) – строка фиксированной длины. При хранении всегда дополняется пробелами в конце строки до заданного размера. Диапазон M: от 0 до 255 симв. При выводе значения концевые пробелы удаляются.

VARCHAR(M) – строка переменной длины, хранится то количество символов, которое необходимо, плюс один байт для записи длины. Диапазон M: от 0 до 255 симв.

Извлеченные из столбцов CHAR(10) и VARCHAR(10) величины одинаковы (из CHAR удаляются концевые пробелы).

Не задан BINARY – величины в столбцах типа CHAR и VARCHAR сортируются и сравниваются без учета регистра, иначе с учетом регистра.

Извлечение данных, оператор SELECT

```
SELECT [TOP n |DISTINCT | DISTINCTROW|ALL]
        select_выражение,...
     [INTO {OUTFILE | DUMPFILE} 'file' options]
 [FROM источник_записей ]
 [WHERE условия_отбора_записей]
 [GROUP BY {атрибут | формула} [ASC | DESC],...]
 [HAVING фильтр_группы]
 [ORDER BY {атрибут | формула} [ASC | DESC],...]
 [LIMIT  колич_строк ]
```

Нотация Бекуса Наура: **[. . .]** – не обязательное предложение, **{a1 | a2} – a1** или **a2.**

Отличия в синтаксисе

Access : **TOP n** – вывод первых **n** записей;

DISTINCTROW, ALL – вывод всех строк;

DISTINCT – вывод уникальных строк;

Пример 1: Таблица **Id_Ip {Id, Ip},** где **Id** - ключ, номер записи и **Ip**- адрес, с которого приходили на сайт.

Select Distinct Ip from Id_Ip; – выборка уникальных **Ip**.

MySQL : нет **TOP n**, вместо него **LIMIT rows;**

DISTINCT, DISTINCTROW – вывод уникальных строк; **ALL** – вывод всех строк.

Разделы (параметры) оператора SELECT

FROM – Определяет источник записей для полей и выражений, перечисленные в разделе **SELECT**.

WHERE – Определяет, какие записи из таблиц, перечисленных в разделе **FROM**, следует включить в результат выполнения оператора **SELECT**. Фактически это фильтр по всем записям.

GROUP BY – Объединяет записи с одинаковыми значениями в указанном списке полей в одну группу.

HAVING – Определяет условия выбора из групп. Фактически это фильтр по записям внутри группы.

ORDER BY – Сортирует записи, полученные в результате запроса, в порядке возрастания или убывания на основе значений указанного поля или полей.

Схема выполнения запроса на выборку. Шаг 1

Шаг 1. **Раздел** **FROM**. Определяет источник записей для выборки оператором **SELECT.** В качестве источника записей могут использоваться таблицы, представления, подзапросы. Если указано несколько источников записей, то на шаге 1 создается их расширенное декартово произведение (назовем его таблица T) . Если в разделе FROM указана только одна таблица, то она же и является результатом выполнения этого раздела. Допускается переименование таблиц, указанных в разделе FROM, при помощи оператора AS.

Выполнение запроса на выборку. Шаг 1

Отношение R

A1	A2
A	1
A	2
B	1
B	3
B	4

text integer

Отношение S

B1	B2
1	h
2	g
3	h

integer text

Шаг1: Декартово произвед.

Select R.a1, R.a2, S.b1, S.b2

From R, S;

Таблица T: декарт. пр. R, S (4 столб.)

A1	A2	B1	B2
a	1	1	h
a	1	2	g
a	1	3	h
a	2	1	h
a	2	2	g
a	2	3	h
b	1	1	h
b	1	2	g
b	1	3	h
b	3	1	h
b	3	2	g
b	3	3	h
b	4	1	h
b	4	2	g
b	4	3	h

Схема выполнения запроса на выборку

Шаг 2. На втором шаге выполняется раздел WHERE. Условное выражение этого раздела применяется к каждой строке таблицы **T**, и результатом является таблица **T1**, содержащая те и только те строки таблицы **T**, для которых результатом вычисления логического выражения является true. (Заголовки таблиц T и T1 совпадают.)

Если раздел WHERE в операторе выборки отсутствует, то это трактуется как наличие раздела WHERE true, т. е. **T1** содержит те и только те строки, которые содержатся в таблице **T**.

Выполнение запроса на выборку. Шаг 2

Таблица T: декарт. произв. R, S

A1	A2	B1	B2
a	1	1	h
a	1	2	g
a	1	3	h
a	2	1	h
a	2	2	g
a	2	3	h
b	1	1	h
b	1	2	g
b	1	3	h
b	3	1	h
b	3	2	g
b	3	3	h
b	4	1	h
b	4	2	g
b	4	3	h

```
SELECT R.a1,
  S.b1, S.b2
FROM R, S
WHERE R.a2=S.b1;
```

Таблица T1 (4 столб)

$R \bowtie_F S$, F=(R.a2=S.b1)			
R.a1	**R.a2**	**S.b1**	**S.b2**
a	1	1	h
a	2	2	g
b	3	3	h
b	1	1	h

Схема выполнения запроса на выборку

Шаг 3. Если в операторе выборки присутствует раздел GROUP BY, то он выполняется на третьем шаге. Каждый элемент списка имен столбцов, указываемого в этом разделе, должен быть одним из имен столбцов таблицы T1 или агрегатной функцией (**`COUNT, SUM, MIN, MAX, AVG`**) от других столбцов.

В результате выполнения раздела GROUP BY образуется сгруппированная таблица T2, в которой строки таблицы T1 расставлены в минимальное число групп, таких, что во всех строках одной группы значения столбцов, указанных в списке имен столбцов раздела GROUP BY (столбцов группировки), одинаковы.

Сгруппированные таблицы не могут являться окончательным результатом оператора выборки. Они существуют только на концептуальном уровне на стадии выполнения запроса, содержащего раздел GROUP BY.

Выполнение запроса на выборку. Шаг 3

Таблица T1 (4 столб)

$R \bowtie_F S$, F=(R.a2=S.b1)			
R.a1	**R.a2**	**S.b1**	**S.b2**
a	1	1	h
a	2	2	g
b	3	3	h
b	1	1	h

```
Select R.a1, R.a2, S.b2
From R, S  Where R.a2=S.b1
Group by R.a1;
```

Таблица T2 (4 столб), сгруппированная по 1 столб.

$R \bowtie_F S$, F=(R.a2=S.b1)			
R.a1	**R.a2**	**S.b1**	**S.b2**
a	1	1	h
a	2	2	g
b	3	3	h
b	1	1	h

Выполнение запроса на выборку. Шаг 3

Таблица T1 (4 столб)

$R \bowtie_F S$, F=(R.a2=S.b1)			
R.a1	**R.a2**	**S.b1**	**S.b2**
a	1	1	h
a	2	2	g
b	3	3	h
b	1	1	h

```
Select R.a1, R.a2, S.b2
From R, S  Where R.a2=S.b1
Group by R.a1, R.a2;
```

Таблица T2 (4 столб), сгруппированная по 2 столб.

$R \bowtie_F S$, F=(R.a2=S.b1)			
R.a1	**R.a2**	**S.b1**	**S.b2**
a	1	1	h
a	2	2	g
b	3	3	h
b	1	1	h

Выполнение запроса на выборку. Шаг 4

Шаг 4. Выполнение раздела HAVING, если в операторе выборки оно присутствует. Условное выражение этого раздела применяется к каждой группе строк таблицы T2, и результатом является сгруппированная таблица T3, содержащая те и только те группы строк таблицы T2, для которых результатом вычисления условного выражения является true. Условное выражение раздела HAVING применяется к группам строк, а не к отдельным строкам. В условном выражении могут использоваться имена столбцов группировки и агрегатные функции (COUNT, SUM, MIN, MAX, AVG) от других столбцов.

Выполнение запроса на выборку. Шаг 4

Шаг 4. При наличии в запросе раздела HAVING, которому не предшествует раздел GROUP BY, таблица T1 рассматривается как сгруппированная таблица, состоящая из одной группы строк, без столбцов группирования. В этом случае логическое выражение раздела HAVING может состоять только из предикатов с агрегатными функциями, а результат вычисления этого раздела T3 либо совпадает с таблицей T1, либо является пустым.

GROUP BY может существовать без HAVING.

HAVING без GROUP BY в Access инициирует ошибку.

Выполнение запроса на выборку. Шаг 4

Таблица T2 (4 столб) с 3 группами

$R \bowtie_F S$, F=(R.a2=S.b1)			
R.a1	**R.a2**	**S.b1**	**S.b2**
a	1	1	h
a	2	2	g
b	3	3	h
b	1	1	h

```
Select R.a1, R.a2, S.b2
From R, S  Where R.a2=S.b1
Group by R.a1, R.a2, S.b2
Having a2>1;
```

Таблица T3 (4 столб), сгруппированная по 3 столб. с выполненным условием для группы

$R \bowtie_F S$, F=(R.a2=S.b1)			
R.a1	**R.a2**	**S.b1**	**S.b2**
a	2	2	g
b	3	3	h

Выполнение запроса на выборку. Шаг 5

Шаг 5.

А) Если нет раздела GROUP BY (с HAVING или без него) –

после выполнения предложения WHERE выполняется предложение SELECT. Строится проекция на основе таблицы T1.

Б) Если есть раздел GROUP BY (нет HAVING) предложение SELECT выполняется на основе сгруппированной таблицы T2

В) Если есть разделы GROUP BY и HAVING) предложение SELECT выполняется на основе сгруппированной и отфильтрованной таблицы T3

Результат – таблица T4.

Количество строк = количество строк в таблице T1 или количество строк = количество групп строк в таблице T3.

Число столбцов в таблице T4 зависит от числа элементов в списке элементов выборки и от вида элементов.

Выполнение запроса на выборку. Шаг 5

Таблица T3 (4 столб) с отфильтрованными данными

$R \bowtie_F S$, F=(R.a2=S.b1)			
R.a1	R.a2	S.b1	S.b2
a	2	2	g
b	3	3	h

```
Select R.a1, R.a2, S.b2
From R, S  Where R.a2=S.b1
Group by R.a1, R.a2, S.b2
Having a2>1;
```

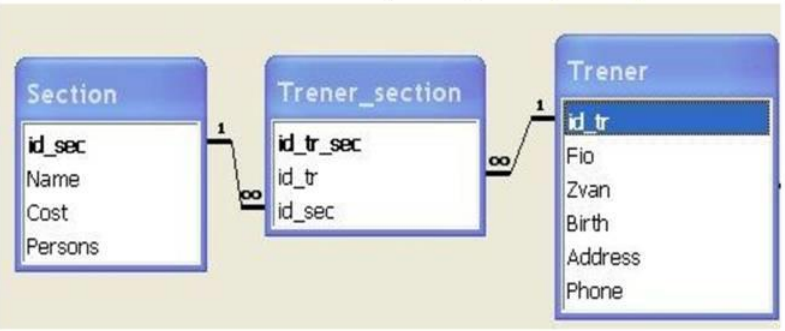
Таблица T4 (3 столб). - проекция

$R \bowtie_F S$, F=(R.a2=S.b1)			
R.a1	R.a2	S.b2	
a	2	g	
b	3	h	

Раздел FROM. Объединение источников записей. Объединение за счет условия в разделе Where.

Допускается объединение источников записей, перечисленных в разделе **From.** В качестве источника записей могут использоваться базовые таблицы, представления, подчиненные запросы

Спортклуб. В спортклубе несколько секций. Один тренер может вести несколько секций и одну секцию может вести несколько тренеров **(M:N)**.



Раздел FROM. Объединение за счет условия в разделе Where.

2 столбика, 4 записи

Section : таблица

	id_sec	Name
+	1	аэробика
+	2	плавание с кругом
+	3	волейбол
+	4	бег на месте

2 столбика, 6 записей

Trener : таблица

	id_tr	Fio
+	2	Иванов И.И.
+	3	дон Румата
+	4	Карась В.Н.
+	5	Максимова
+	10	Чирик З.П.
+	11	Оболенский

3 столбика, 6 записей

Trener_section : таблица

	id_tr_sec	id_tr	id_sec
+	1	3	2
+	2	3	4
+	3	2	1
+	4	5	1
+	5	4	3
+	6	5	3

Раздел FROM. Объединение без условия в Where.

Пример 4. `SELECT * from Trener,Section;`

`Trener (2 ст,6 зап)`
`Section (2 ст,4 зап)` → Результат: расширенное декартово произведение **24 зап., 4 столб.**

Пример 5. `SELECT * from Trener, Trener_Section;`

`Trener (2 ст,6 зап)`
`Trener_Section (3 ст,6 зап)` → Результат: расширенное декартово произведение **36 зап., 5 столб.**

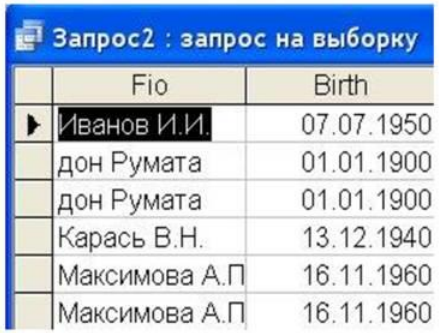
Пример 6. `SELECT * from Trener,Section, Trener_Section;`

`Trener (2 ст,6 зап)`
`Section (2 ст,4 зап)` → Результат: расширенное декартово произведение **6*4*6 = 144 зап., 7 столб.**
`Trener_Section (3 ст,6 зап)`

Раздел FROM. Объединение с условием в Where.

Пример 7.

```
SELECT Fio, Birth
From Trener, Trener_Section
Where (Trener.id_tr =
Trener_Section.id_tr) ;
```


Запрос2 : запрос на выборку

Fio	Birth
Иванов И.И.	07.07.1950
дон Румата	01.01.1900
дон Румата	01.01.1900
Карась В.Н.	13.12.1940
Максимова А.П	16.11.1960
Максимова А.П	16.11.1960

Пример 8.

```
SELECT Distinct Fio,
Birth
From Trener,
Trener_Section
Where (Trener.id_tr =
Trener_Section.id_tr) ;
```


Запрос2 : запрос на выборку

Fio	Birth
дон Румата	01.01.1900
Иванов И.И.	07.07.1950
Карась В.Н.	13.12.1940
Максимова А.П	16.11.1960

Раздел FROM. Объединение с условием в Where.

Пример 9. К условию объединения добавим ограничение по возрасту

```
SELECT Fio, Birth
From Trener, Trener_Section
Where (Trener.id_tr =
Trener_Section.id_tr)      and
(Year(Birth)>1940) ;
```

Раздел FROM. Объединение с условием в Where.

Пример 10. Сформировать список тренеров и названия секций, которые они ведут.

```
SELECT  Fio, Name
From Trener, Section, Trener_Section
  Where (Trener.id_tr=Trener_Section.id_tr)
  And (Section.id_sec=Trener_Section.id_sec);
```

Запрос2 : запрос на выборку

Fio	Name
дон Румата	плавание с кругом
дон Румата	бег на месте
Иванов И.И.	аэробика
Максимова А.П	аэробика
Карась В.Н.	волейбол
Максимова А.П	волейбол

Объединение таблицы сама с собой

Применяется в рекурсивных запросах.

Объединение таблицы с собой − объединение двух копий одной и той же таблицы. При объединении таблицы с собой, все одинаковые имена столбца дополняются префиксами имени таблицы. Чтобы ссылаться к столбцам внутри запроса, − нужно иметь два различных имени для таблицы.

Псевдонимы = переменные диапазона = переменные корреляции или просто "псевдонимами".

Псевдонимы используются в разделах Select и From.

Определение псевдонимов таблицы в предложении FROM. MySQL − имя таблицы, пробел, псевдоним.

Access - имя таблицы, AS, псевдоним.

Объединение таблицы сама с собой

Таблица сотрудников отдела, должности, подчинение.

Sotrudniki : таблица

	Имя поля	Тип данных
🔑	id_sotr	Счетчик
	FIO	Текстовый
	dolg	Текстовый
	podch	Числовой

Sotrudniki : таблица

	id_sotr	FIO	dolg	podch
	1	Степанов Б.И.	зав.отделом	0
	2	Иванов В.А.	рук.группы разработчиков	1
	3	Кулик.В.П.	вед.программист	2
	4	Колесова Н.В.	млад.программист	2
	5	Иванченко Д.С.	ст.лаборант	2
	6	Ивановская В.Д.	рук.группы сопровождения	1
	7	Куликов.В.П.	вед.программист	6
	8	Колесов В.В.	млад.программист	6
	9	Иваненко Г.С.	ст.лаборант	6
▶	Счетчик)			0

Запрос на выборку

Sotrudniki : таблица

	id_sotr	FIO	dolg	podch
	1	Степанов Б.И.	зав.отделом	0
	2	Иванов В.А.	рук.группы разработчиков	1
	3	Кулик.В.П.	вед.программист	2
	4	Колесова Н.В.	млад.программист	2
	5	Иванченко Д.С.	ст.лаборант	2
	6	Ивановская В.Д.	рук.группы сопровождения	1
	7	Куликов.В.П.	вед.программист	6
	8	Колесов В.В.	млад.программист	6
	9	Иваненко Г.С.	ст.лаборант	6
▶	Счетчик)			0

Пример 2: Сформировать сведения о сотрудниках отдела и их должностях, а также ФИО и должность непосредственного начальника каждого сотрудника.

```
SELECT Pod.Fio, Pod.id_sotr, Pod.dolg, Ruk.fio,
Ruk.dolg, Ruk.id_sotr
FROM Sotrudniki as Ruk, Sotrudniki as Pod
where Pod.podch = Ruk.id_sotr;
```

Запрос на выборку сотрудников. Шаг 1

```
SELECT  Pod.Fio,  Pod.id_sotr,  Pod.dolg,
Ruk.fio, Ruk.dolg, Ruk.id_sotr
FROM Sotrudniki as Ruk, Sotrudniki as Pod
where Pod.podch = Ruk.id_sotr;
```

Ruk.Id_sotr	Ruk.FIO	Ruk.dolg	Ruk.podch	Pod.id_sotr	Pod.FIO	Pod.dolg	Pod.podch
1	Степанов	Зав.отделом	0	1	Степанов	Зав.отделом	0
1	Степанов	Зав.отделом	0	2	Иванов	Рук.гр.	1
1	Степанов	Зав.отделом	0	3	Кулик	Вед.пр.	2
….	….	…	..	…	…	….	….
1	Степанов	Зав.отделом	0	9	Иваненко	Ст.лаб	6
2	Иванов	Рук.гр.	1	680×289	Степанов	Зав.отделом	0
2	Иванов	Рук.гр.	1	2	Иванов	Рук.гр.	1
…	..	..	…	..	…	…	…
2	Иванов	Рук.гр.	1	9	Иваненко	Ст.лаб	6
…	….	….	….	….	…	….	….
9	Иваненко	Ст.лаб	6	1	Степанов	Зав.отделом	0
….	….	….	…	…	…	…	…
9	Иваненко	Ст.лаб	6	9	Иваненко	Ст.лаб	6

Запрос на выборку сотрудников. Шаг 2

Условие **Pod.podch = Ruk.id_sotr**

Ruk.Id_sotr	Ruk.FIO	Ruk.dolg	Ruk.podch	Pod.id_sotr	Pod.FIO	Pod.dolg	Pod.podch
1	Степанов	Зав.отделом	0	1	Степанов	Зав.отделом	0
1	Степанов	Зав.отделом	0	2	Иванов	Рук.гр.	1
1	Степанов	Зав.отделом	0	3	Кулик	Вед.пр.	2
....	....	...	..	...	...	....	....
1	Степанов	Зав.отделом	0	9	Иваненко	Ст.лаб	6
2	Иванов	Рук.гр.	1	680к289	Степанов	Зав.отделом	0
2	Иванов	Рук.гр.	1	2	Иванов	Рук.гр.	1
...	..	..	...	..	...	...	...
2	Иванов	Рук.гр.	1	9	Иваненко	Ст.лаб	6
...	....	....	....	....	...	....	....
9	Иваненко	Ст.лаб	6	1	Степанов	Зав.отделом	0
....	....	....	...	...	...	...	...
9	Иваненко	Ст.лаб	6	9	Иваненко	Ст.лаб	6

Запрос на выборку сотрудников. Результат

ZSotrud : запрос на выборку

Pod.fio	Po	Pod.dolg	Ruk.fio	Ruk.dolg	Ruk
Иванов В.А.	2	рук.группы разработчиков	Степанов Б.И.	зав.отделом	1
Кулик.В.П.	3	вед.программист	Иванов В.А.	рук.группы разработчиков	2
Колесова Н.В.	4	млад.программист	Иванов В.А.	рук.группы разработчиков	2
Иванченко Д.С.	5	ст.лаборант	Иванов В.А.	рук.группы разработчиков	2
Ивановская В.Д.	6	рук.группы сопровождения	Степанов Б.И.	зав.отделом	1
Куликов.В.П.	7	вед.программист	Ивановская В.Д.	рук.группы сопровождения	6
Колесов В.В.	8	млад.программист	Ивановская В.Д.	рук.группы сопровождения	6
Иваненко Г.С.	9	ст.лаборант	Ивановская В.Д.	рук.группы сопровождения	6