Подзапросы и предикаты

Проблема

- Требование реляционной модели – суперпозиция реляционных операций
- SELECT обеспечивает ее лиши частично

Решение

- Раннее решение – использовать представления

```
CREATE  [OR ALTER] VIEW <name_view>
        [ (<nm_col1> [,<nm_col2>, …)]
 AS <select_operator>

DROP VIEW <name_view>
```

- В операторе SELECT нет различий между таблицами и представлениями

Подзапрос

- Подзапрос – это оператор SELECT, включенный в спецификацию другого оператора SQL
- Если подзапрос ссылается на поля из таблиц в основном запросе, то он называется кореллированным

Использование подзапросов

- В операторе INSERT

INSERT INTO <*таблица*> [(<*список столбцов*>)]

(SELECT ...)

Использование подзапросов

- для формирования выходного столбца основного оператора SELECT

```
SELECT e.FULL_NAME, e.SALARY,
(SELECT MAX (m.SALARY)
        FROM EMPLOYEE m
        WHERE m.JOB_COUNTRY='USA') AS
   MAX_SALARY_USA
FROM EMPLOYEE e
WHERE e.JOB_COUNTRY='USA'
```

Использование подзапросов

- в вычислимом столбце

```
SELECT e.FULL_NAME, e.SALARY,
(SELECT MAX (m.SALARY)
        FROM EMPLOYEE m
        WHERE m.JOB_COUNTRY='USA') - e.SALARY
                AS  DIFF_SALARY_USA
FROM EMPLOYEE e
WHERE e.JOB_COUNTRY='USA'
```

Использование подзапросов

```
SELECT  FULL_NAME, SALARY,
(SELECT CURRENCY FROM COUNTRY  c
   WHERE
c.COUNTRY=e.JOB_COUNTRY) AS CURRENCY
FROM EMPLOYEE e
```

Такой подзапрос называется коррелированным.

Это неявная форма операции соединения

Использование подзапросов

- в предложении FROM (начиная с версии Firebird 2.0)
- в этом случае подзапрос выступает в роли «производной таблицы» (*derived table*)

Derived table

(select-запрос)

[[AS] алиас производной таблицы]

[(<псевдонимы полей производной таблицы>)]

Derived table

```
SELECT  P.PROJ_NAME, T.DEPT_NAME
    FROM
      PROJECT P LEFT JOIN

(SELECT PD.PROJ_ID,  D.DEPARTMENT
      FROM PROJ_DEPT_BUDGET PD
                JOIN DEPARTMENT D
                        ON (PD.DEPT_NO=D.DEPT_NO)
       WHERE PD.FISCAL_YEAR=1995)

AS T ( PROJ_ID, DEPT_NAME)
ON   (P.PROJ_ID=T.PROJ_ID)
```

Альтернатива - представление

```
CREATE VIEW DT ( PROJ_ID, DEPT_NAME)
AS
SELECT PD.PROJ_ID,  D.DEPARTMENT
      FROM PROJ_DEPT_BUDGET PD
                JOIN DEPARTMENT D
                        ON (PD.DEPT_NO=D.DEPT_NO)
       WHERE PD.FISCAL_YEAR=1995;

SELECT  P.PROJ_NAME, T.DEPT_NAME
    FROM      PROJECT P LEFT JOIN DT ON
          (P.PROJ_ID=DT.PROJ_ID)
```

Использование подзапросов

- для получения значений или условий, используемых в предикатах поиска предложения WHERE операторов SELECT, UPDATE и DELETE, а также в предложении HAVING для группирующего запроса

- Если подзапрос возвращает одно значение (скалярный запрос), его можно использовать в θ-условии для сравнения на равенство или неравенство

```
SELECT FULL_NAME
   FROM EMPLOYEE  a
 WHERE
      a.SALARY >
      (SELECT AVG(b.SALARY) FROM EMPLOYEE b);
```

- Если подзапрос возвращает таблицу, содержащую один столбец и произвольное количество строк, для использования его в подзапросе используются предикаты
- В этом случае возвращаемый столбец рассматривается как множество

Предикат IN

```
SELECT FULL_NAME
    FROM EMPLOYEE
    WHERE DEPT_NO
            IN
        (SELECT DEPT_NO
                FROM DEPARTMENT
                WHERE MNGR_NO IS NULL)
```

Предикаты ALL, SOME и ANY

```
SELECT PROJ_ID, DEPT_NO, FISCAL_YEAR
        FROM PROJ_DEPT_BUDGET
        WHERE
                PROJECTED_BUDGET>
                ALL
                   (SELECT BUDGET
                        FROM DEPARTMENT)
```

```
SELECT e.EMP_NO, e.FULL_NAME, e.HIRE_DATE
    FROM EMPLOYEE e
    WHERE e.HIRE_DATE+365>
          SOME
                (SELECT sh.CHANGE_DATE
                      FROM SALARY_HISTORY sh
                      WHERE sh.EMP_NO = e.EMP_NO)

-- возможно  использование ANY
```

Предикат SINGULAR (NOT SINGULAR)

- проверяет, возвращает ли подзапрос в точности один кортеж

Если возвращается NULL или более одного кортежа, то SINGULAR возвращает ложь

Значение предиката не зависит от количества столбцов в подзапросе

```
SELECT FULL_NAME
        FROM EMPLOYEE e
                WHERE
                SINGULAR (SELECT *
                        FROM SALES s
                        WHERE s.SALES_REP=e.EMP_NO);
```

Предикат EXISTS (NOT EXISTS)

- определяет, существует ли (или нет), по крайней мере, один кортеж в выходном результате подзапроса
- Значение предиката не зависит от количества столбцов в подзапросе

```
SELECT a.FULL_NAME
    FROM EMPLOYEE a
    WHERE
        EXISTS (SELECT 1
                    FROM PROJECT p
                    WHERE
                    p.TEAM_LEADER = a.EMP_NO)
```

```
select   d.location
     from department d
      group by location
       having count(*)>=
                  All (select count(*)
                              from department
                              group by location )
```

План выполнения запроса

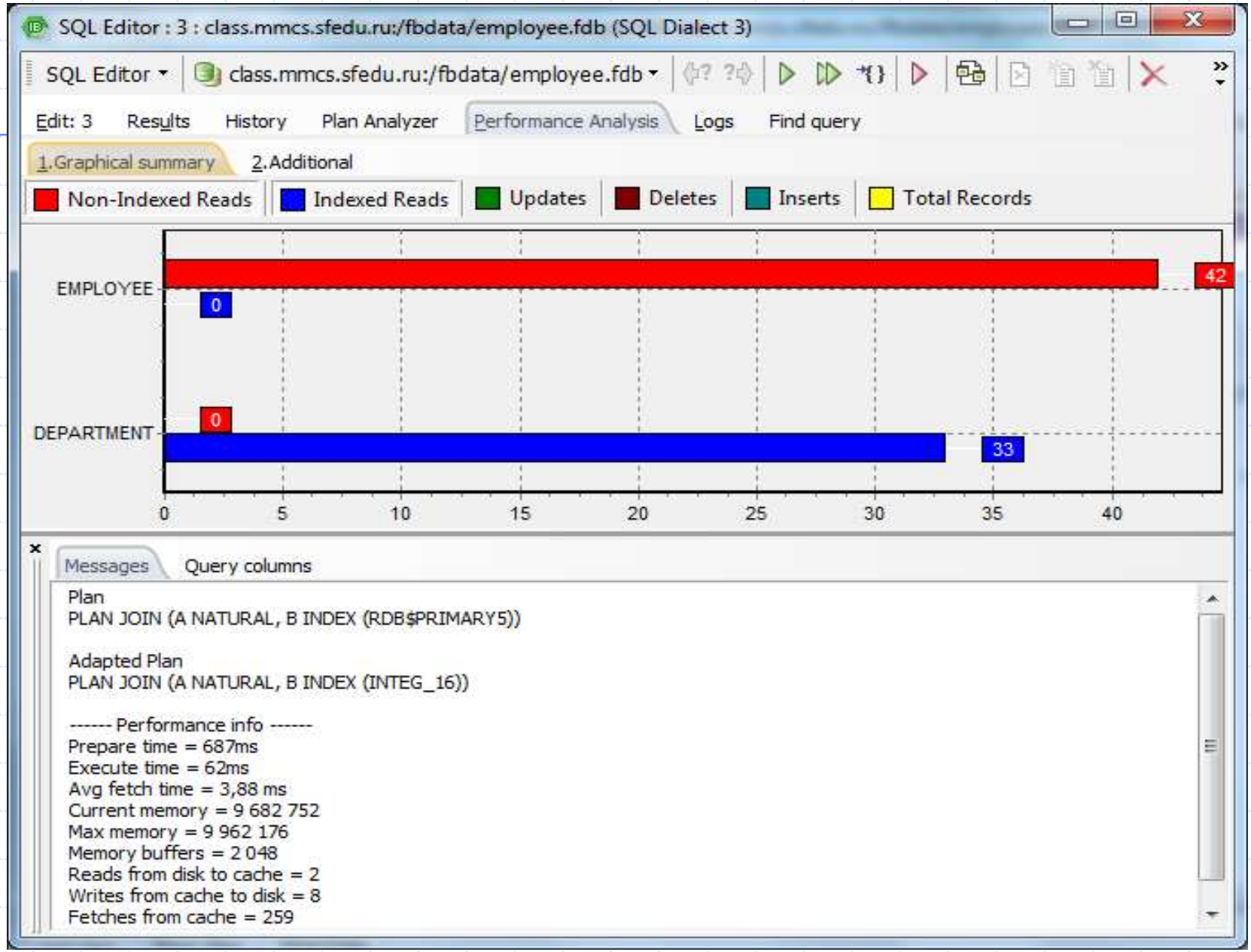
- **План выполнения запроса** — последовательность операций, необходимых для получения результата SQL-запроса в реляционной СУБД

- План в целом разделяется на две стадии:
  - Выборка результатов
  - Сортировка и группировка, выполнение агрегаций

- *Сортировка и группировка* — это опциональная стадия, которая выполняется, если не найдено путей доступа для получения результата в запрошенном порядке.

```
select *
from
  employee A join department B
      on (A.dept_no =B.dept_no)
where job_country ='USA'
```

PLAN JOIN (A NATURAL, B INDEX (RDB$PRIMARY5))


SQL Editor : 3 : class.mmcs.sfedu.ru:/fbdata/employee.fdb (SQL Dialect 3)
SQL Editor
class.mmcs.sfedu.ru:/fbdata/employee.fdb
Edit: 3
Results
History
Plan Analyzer
Performance Analysis
Logs
Find query
1.Graphical summary
2.Additional
Non-Indexed Reads
Indexed Reads
Updates
Deletes
Inserts
Total Records
EMPLOYEE
DEPARTMENT
0
42
0
33
0
5
10
15
20
25
30
35
40
Messages
Query columns
Plan
PLAN JOIN (A NATURAL, B INDEX (RDB$PRIMARY5))
Adapted Plan
PLAN JOIN (A NATURAL, B INDEX (INTEG_16))
------ Performance info ------
Prepare time = 687ms
Execute time = 62ms
Avg fetch time = 3,88 ms
Current memory = 9 682 752
Max memory = 9 962 176
Memory buffers = 2 048
Reads from disk to cache = 2
Writes from cache to disk = 8
Fetches from cache = 259

PLAN <выражение>

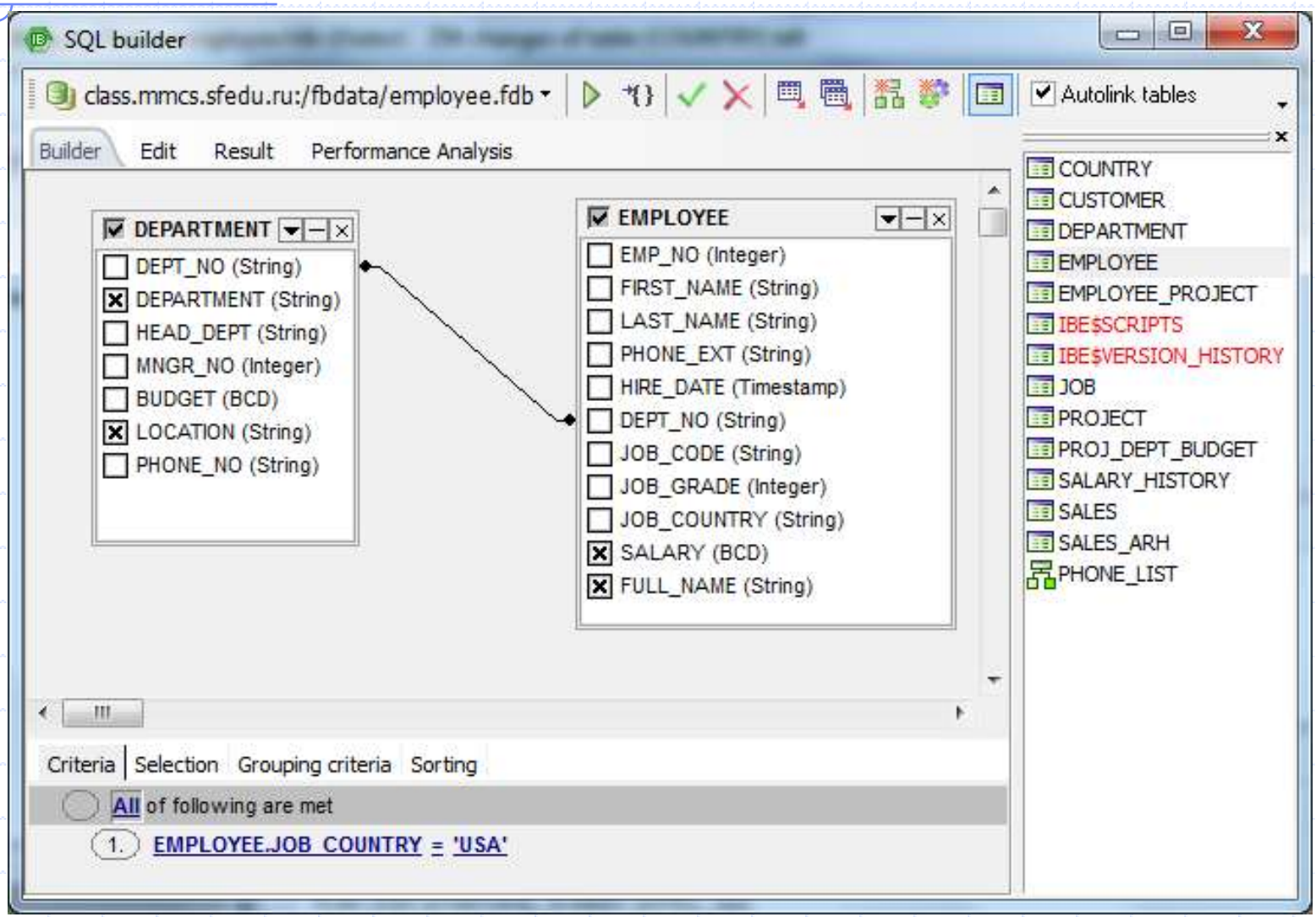
<выражение> ::= [JOIN | [SORT] [MERGE]]
(<элемент> | <выражение>
[, <элемент> | <выражение> ...])

<элемент> ::= {таблица | псевдоним}
{NATURAL
| INDEX (индекс [, индекс ...])
| ORDER индекс [INDEX (индекс [, индекс...])]}

Включение плана в запрос

SELECT [DISTINCT | ALL]
[FIRST <record_number> }|SKIP <record_number>]
<select_list>
FROM <reference_expression_list>
[WHERE <search condition>]
[GROUP BY <group_value_list>
[HAVING <group_condition>]]
[UNION <select operator>]
[PLAN <plan_item_list>]

Построитель запросов



```
select
    department.department,
    department.location,
    employee.full_name,
    employee.salary
from employee
   inner join department on (employee.dept_no =
department.dept_no)
where
        (employee.job_country = 'USA')
```