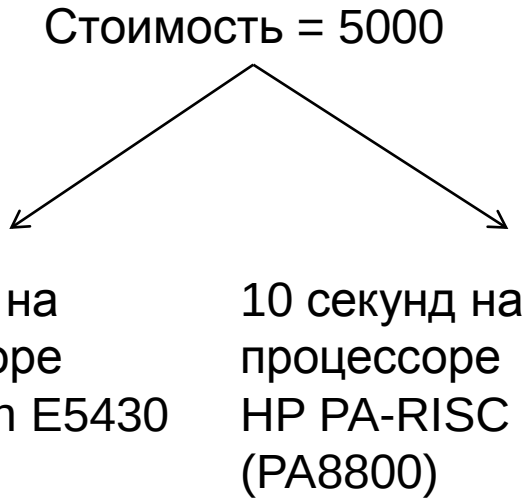
Борчук Леонид

Настройка и самонастройка реляционных систем баз данных при оптимизации обработки запросов

Стоимость выполнения запроса

Стоимость в СУБД Oracle

Cost= 1/sreadtim *
(#SRds * sreadtim +
#MRds * mreadtim
#CPUCycles / cpuspeed),



sreadtim – время одноблочного чтения;

mreadtim – время многоблочного чтения;

cpuspeed – количество тактов за секунду..

#SRds – количество одноболочных чтений с диска;

#MRds – количество многоболочных чтений с диска;

CPUCycles – количество тактов ЦПУ;

План выполнения запроса

Пример (Матиас Ярке, Юрген Кох): служащие, предлагающие компьютерные лекции отделам географически распределенной организации.

employees (enr, ename, status, city)
papers (enr, title, year)
departments (dname, city, street address)
courses (cnr, cname, abstract)
lectures (cnr, dname, enr, daytime)

Названия отделов, расположенных в Нью-Йорке и предлагающих курсы по управлению базами данных

Стратегии выполнения запроса:

Стратегия 1

1. Декартово произведение (Стоимость 200000)
2. Фильтрация (Стоимость 0)

Итого: 200000

Стратегия 2

1. Выполнить слияние (Стоимость 50 + 200)
2. Отсортировать результат (Стоимость 50*200/2.5)
3. Выполнить слияние (Стоимость 400+20)
4. Фильтрация (Стоимость 50*200*2/2.5)

Итого: 6000

Стратегия 3

1. Выполнить слияние (Стоимость 50 + 200)
2. Фильтрация (Стоимость 250)
3. Отсортировать результат (Стоимость 252)
3. Выполнить слияние (Стоимость 20)

Итого: 277

Задача настройки системы баз данных

Для обеспечения требуемого времени ответа необходимы:

1. Адекватные оценки стоимости.

2. Присутствие хотя бы одного плана выполнения с требуемым временем ответа в пространстве состояний.

Распространение ошибок оценок

Стратегия 2

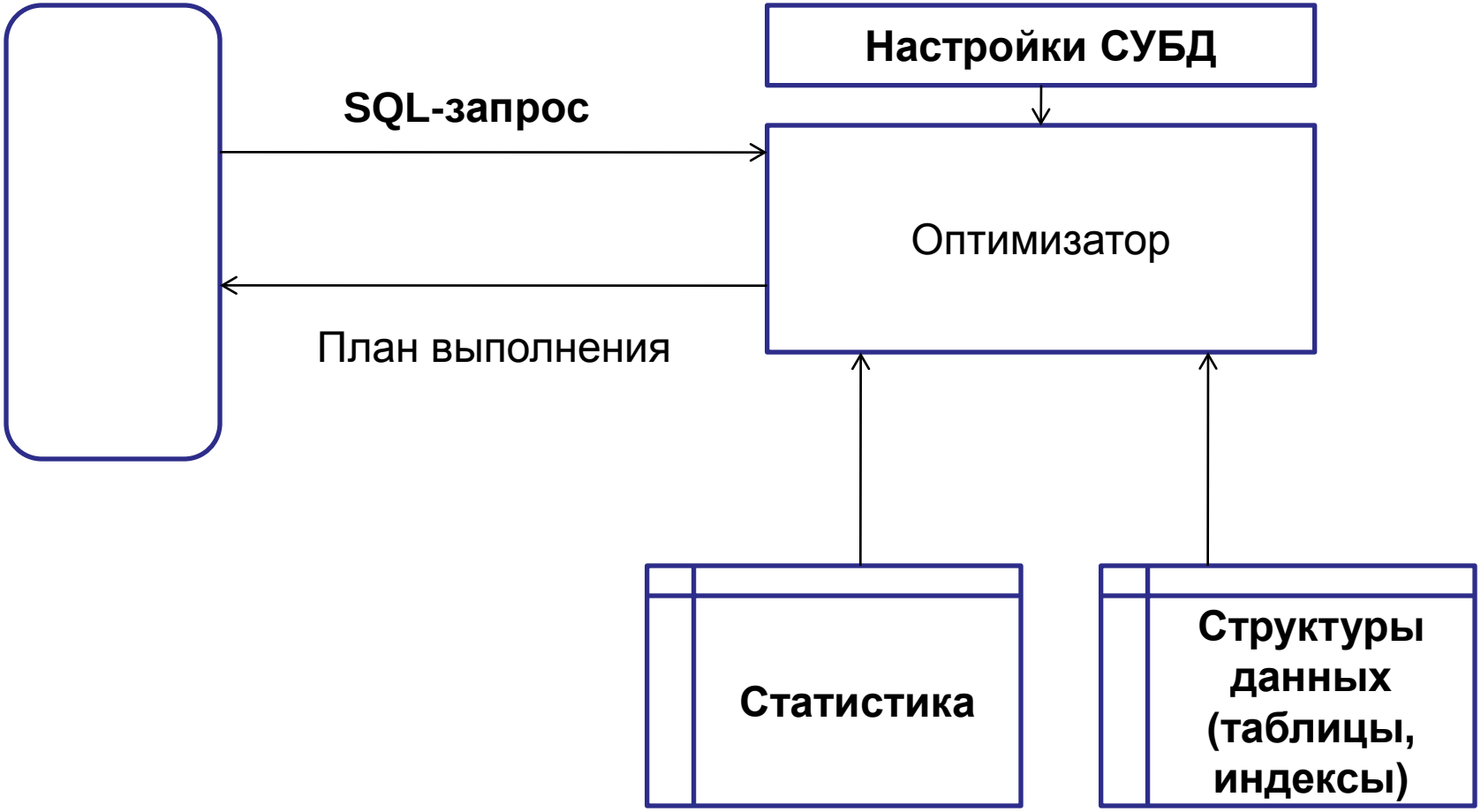
Операция	Стоимость	Ошибочная стоимость
Слияние отношений "courses" и "lectures"	r: 50 + 200 w: 50*200/25 = 400	r: 50 + **100** w: 50*100/25 = 200
Отсортировать результат по dnames	r+w: 400*log(2)400 = 3457	r+w: 200*log(2)200 = 1528
Слияние с отношением "departments"	r: 400 + 20 w: 400 + 400	r: 200 + 20 w: 200 + 200
Фильтрация	r: 800	r:400
Итого (r+w)	6000	2850

Различные варианты написания запроса

Задания второй студенческой олимпиады Oracle (21 команда)

Задание	Количество предложенных вариантов решения
Отбор документов по ключевым словам	15
Замена пустого значения на значение из предыдущей строки	11
Мониторинг потребляемого пространства базы данных	6
Гарантированный выбор	9
Построение дерева объектов компании	5
Выбрать сотрудников с минимальным окладом	6
Посчитать сотрудников, которые никем не руководят	8
Выбор сотрудников, менявших должность без записи в историю	9

Пути настройки системы баз данных



Средства настройки систем баз данных

Demo Case Plan

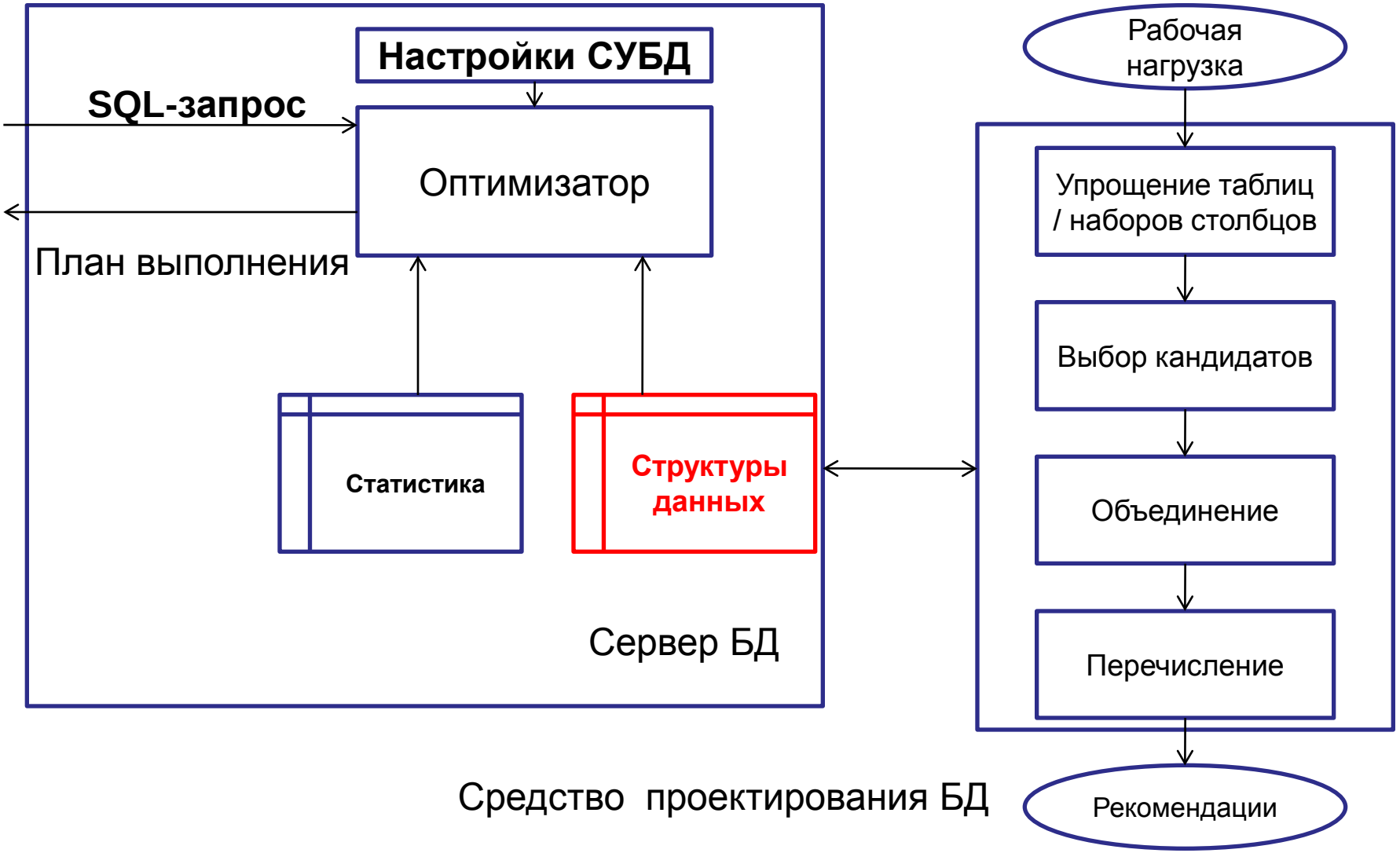
	Rows	card	operation
		2	SELECT STATEMENT
	2	2	SORT GROUP BY
	6,274		FILTER
504.6	13,120	26	HASH JOIN
534.9	208,620	390	HASH JOIN
1.0	15	15	TABLE ACCESS FULL PS_RETROPAYPGM_TBL
858.1	44,621	52	NESTED LOOPS
353.3	14,131	40	HASH JOIN
1.0	5	5	TABLE ACCESS FULL PS_PAY_CALENDAR
3.0	40,000	13,334	TABLE ACCESS FULL WB_JOB
1.6	44,621	27,456	TABLE ACCESS BY INDEX ROWID WB_RETROPAY_EARNS
2.7	74,101	27,456	INDEX RANGE SCAN WB0RETROPAY_EARNS
1.0	13,679	13,679	TABLE ACCESS FULL PS_RETROPAY_RQST
	9,860	1	SORT AGGREGATE
	4,930	1	FIRST ROW
	4,930	1	INDEX RANGE SCAN (MIN/MAX) WB_JOB
	20,022	1	SORT AGGREGATE
	7,750	1	FIRST ROW
	10,011	1	INDEX RANGE SCAN (MIN/MAX) WB_JOB

Самонастраиваемые базы данных

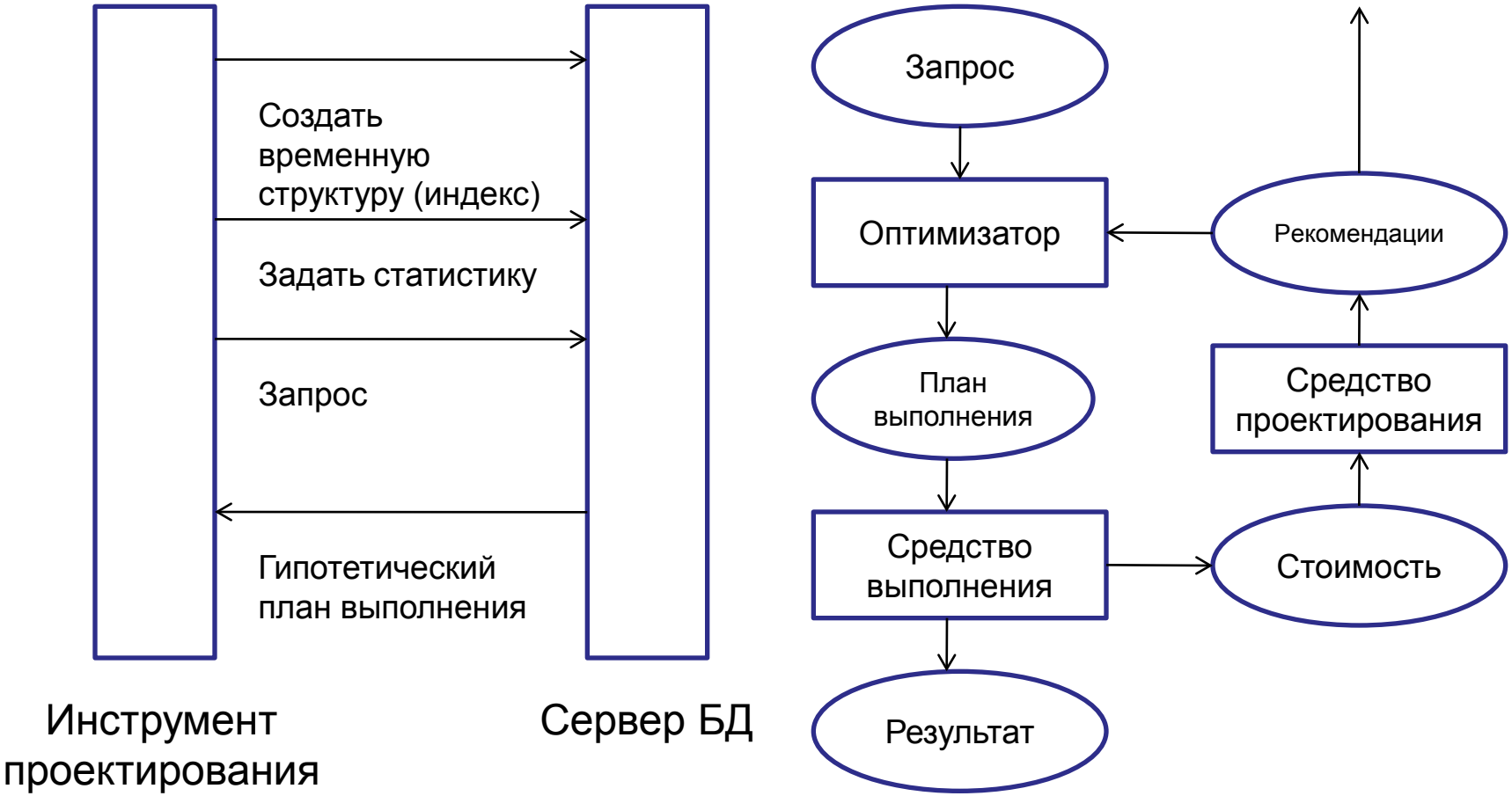
Средства автоматической настройки позволяют:

- производить выработку рекомендаций для построения индексов и материализованных представлений (по рабочей нагрузке);
- производить отбор и воспроизведение рабочей нагрузки;
- автоматически информировать о проблемах производительности;
- производить автоматическое управление статистикой;
- использовать самонастраиваемые гистограммы.

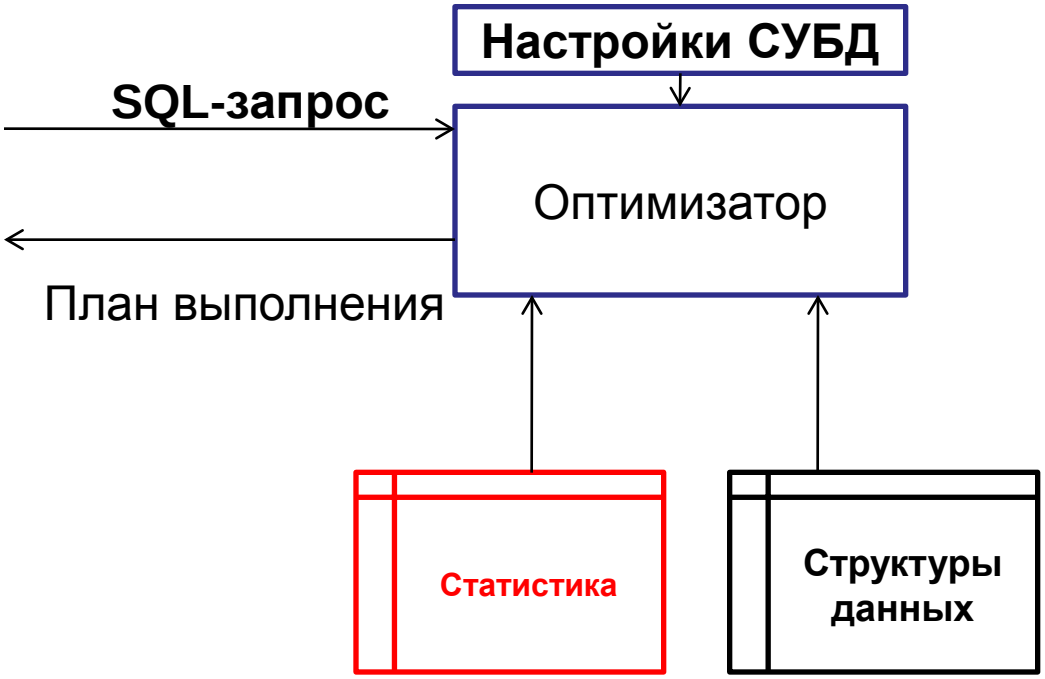
Создание индексов и других структур данных



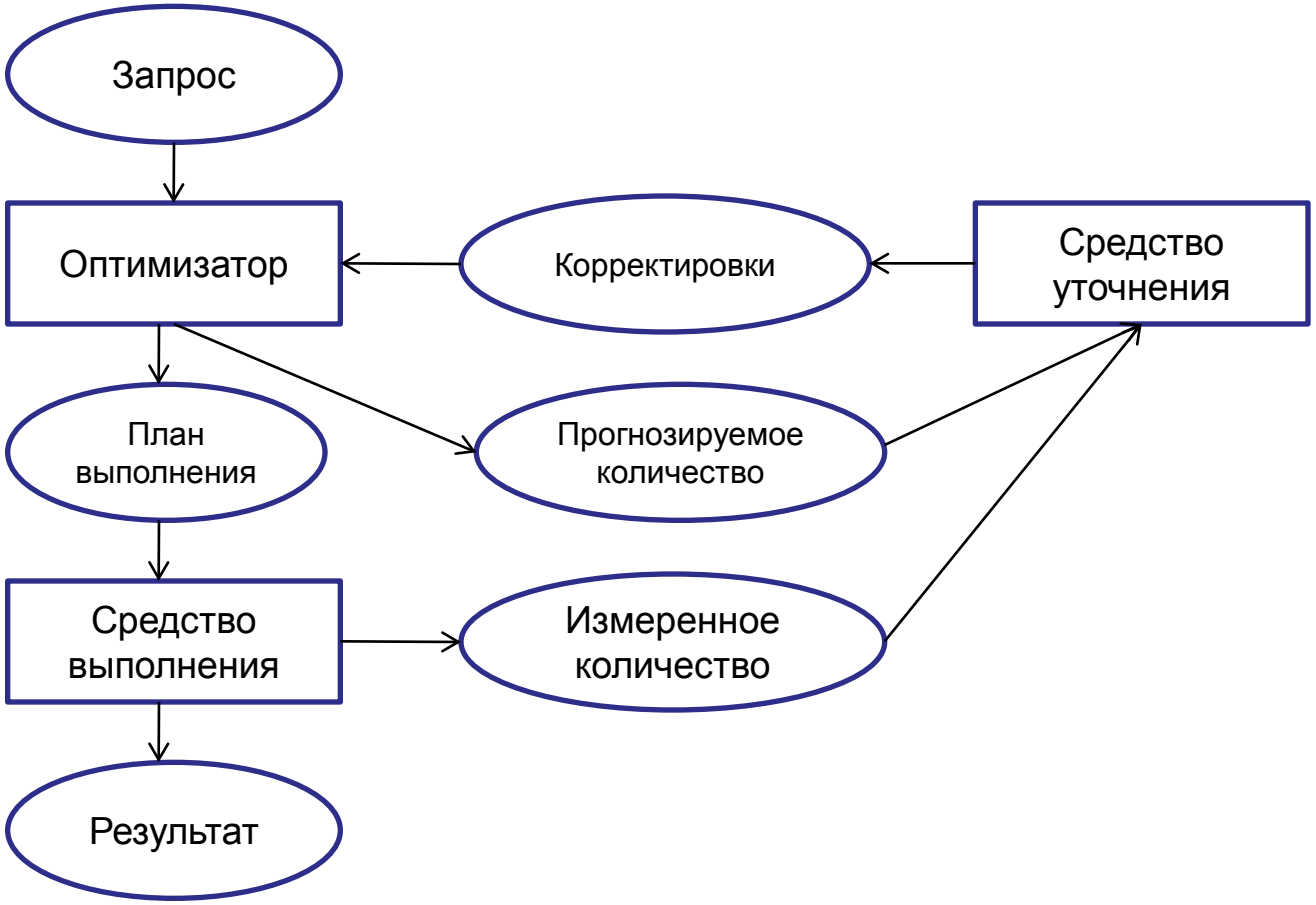
Использование инструментов автоматического проектирования



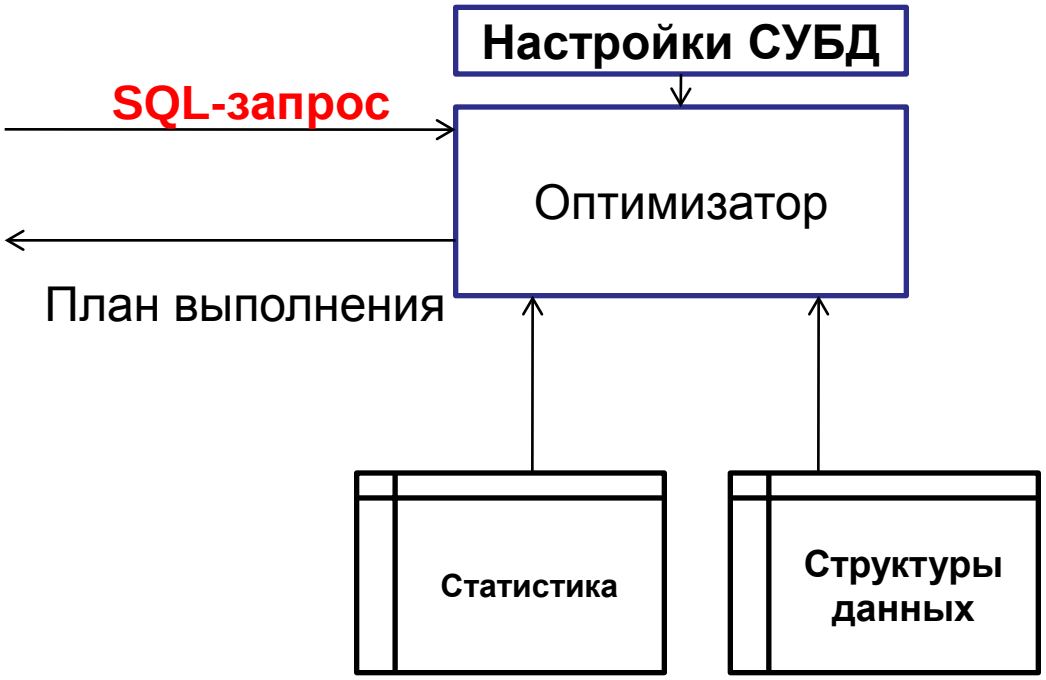
Уточнение статистики

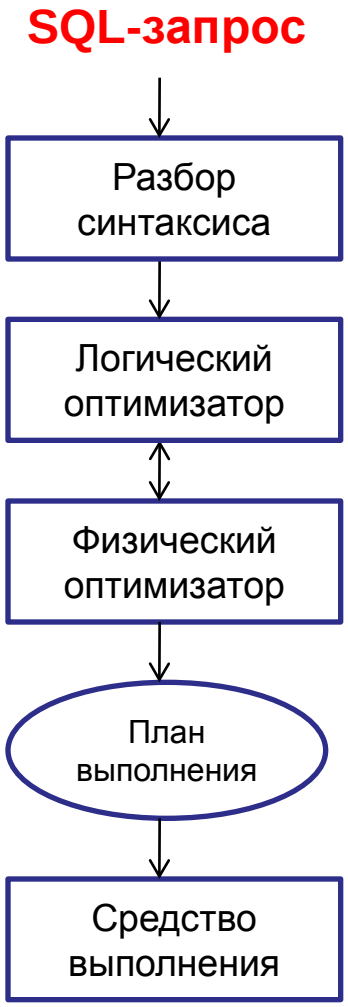


Самонастраиваемые гистограммы



Логические преобразования запроса





Преобразование устранение вложенности

Cost = N*100

N=1, Cost = 100
N = 100, Cost = 10 000
N = 1000, Cost = 100 000

```
Select /*+ qb_name (e1_outer) */ * from
 emp e1 where
 e1.hire_date > sysdate - (10*365)  and
 salary >
     (select/*+ qb_name (e2_inner) */
          avg(salary)  from
            emp e2, dept d1
       where e1.dept_id = e2.dept_id and
               e2.dept_id = d1.dept_id and
        exists
     (select /*+ qb_name (l1_inner) */
            1 from locations l1
     where l1.location_id=d1.location_id ))
```

Cost=100

Cost = 10 000 + N

N=1, Cost = 10001
N = 100, Cost = 10 100
N = 1000, Cost = 11000

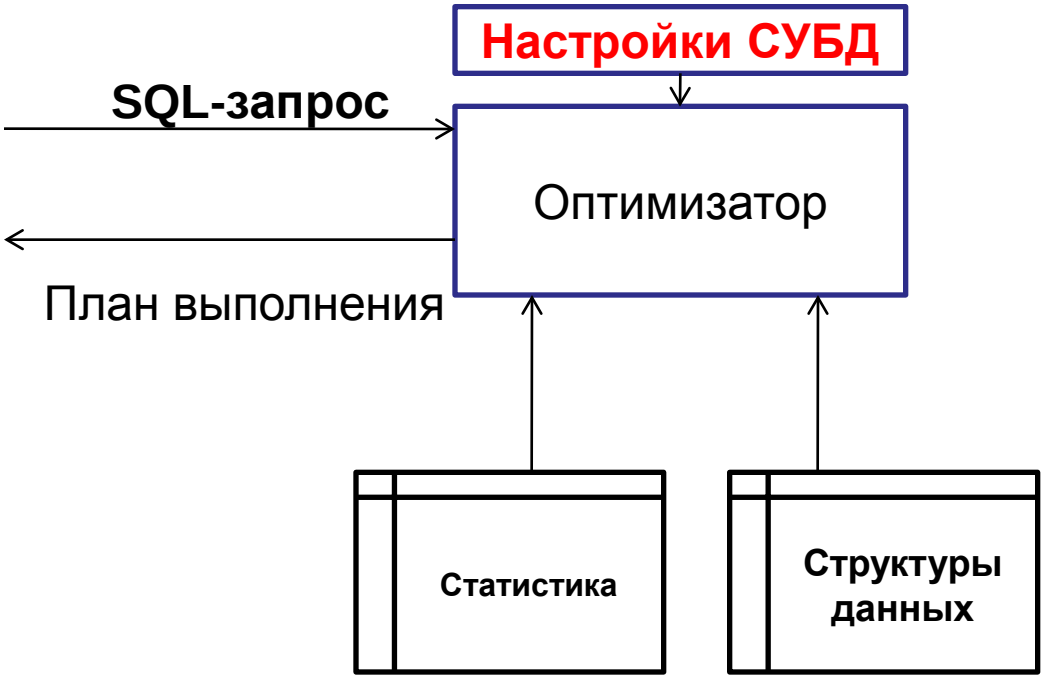
```
Select /*+ qb_name (e1_outer) */ * from
emp e1,
(select /*+ qb_name (e2_inner) */
    dept_id, avg(salary) avg_salary
     from emp e2, dept d1
where e1.dept_id = e2.dept_id and
        e2.dept_id = d1.dept_id and
 exists
     (select /*+ qb_name (l1_inner) */
            1 from locations l1
     where l1.location_id=d1.location_id
     group by dept_id ) gbp1
where
e1.hire_date > sysdate - (10*365)  and
e1. salary > gbp1.avg_salary
```

Cost=10000

Преобразования запросов

1. общая факторизация подвыражений (common sub-expression factorization),
2. слияние представлений "селекция-проекция-соединение" (SPJ view merging),
3. устранение соединений (join elimination),
4. устранение вложенности подзапросов (subquery unnesting),
5. слияние представлений с группировкой (group-by (distinct) view merging),
6. упрощение группировки (group pruning),
7. перемещение предикатов (predicate move around),
8. преобразование операций над множествами в соединения (set operator into join),
9. размещение группировки (group-by placement),
10. вытягивание предиката (predicate pullup),
11. факторизация соединений (join factorization),
12. преобразование дизъюнкции в объединение (disjunction into union-all),
13. преобразование "звезда" (star transformation),
14. проталкивание предикатов соединения (join predicate pushdown).

Уменьшение количества ручек настройки

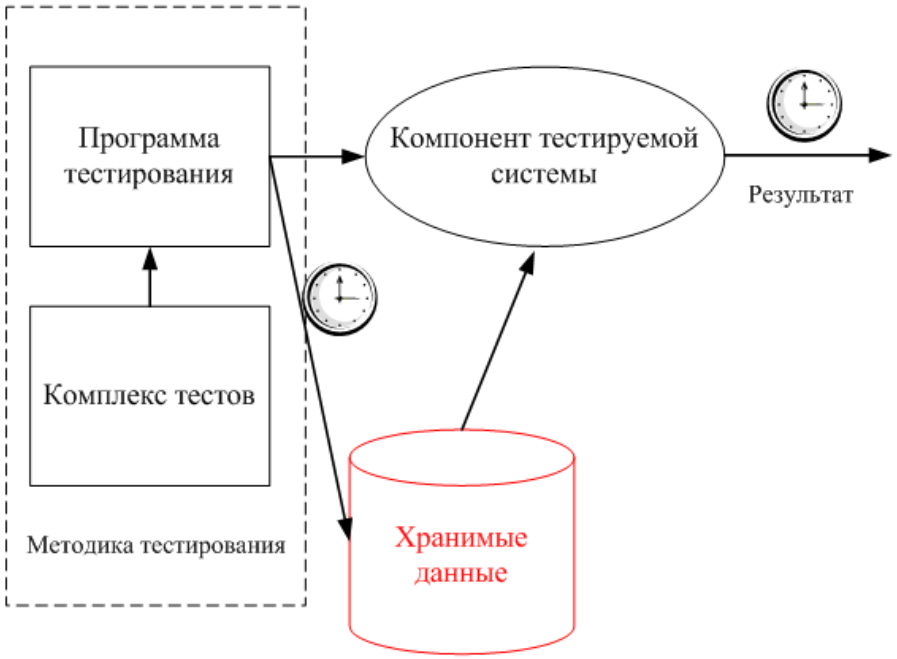


Метод анализа плана выполнения SQL-запроса, используя *O*-большое асимптотики для оценки стоимости

Недостатки подходов к анализу плана выполнения

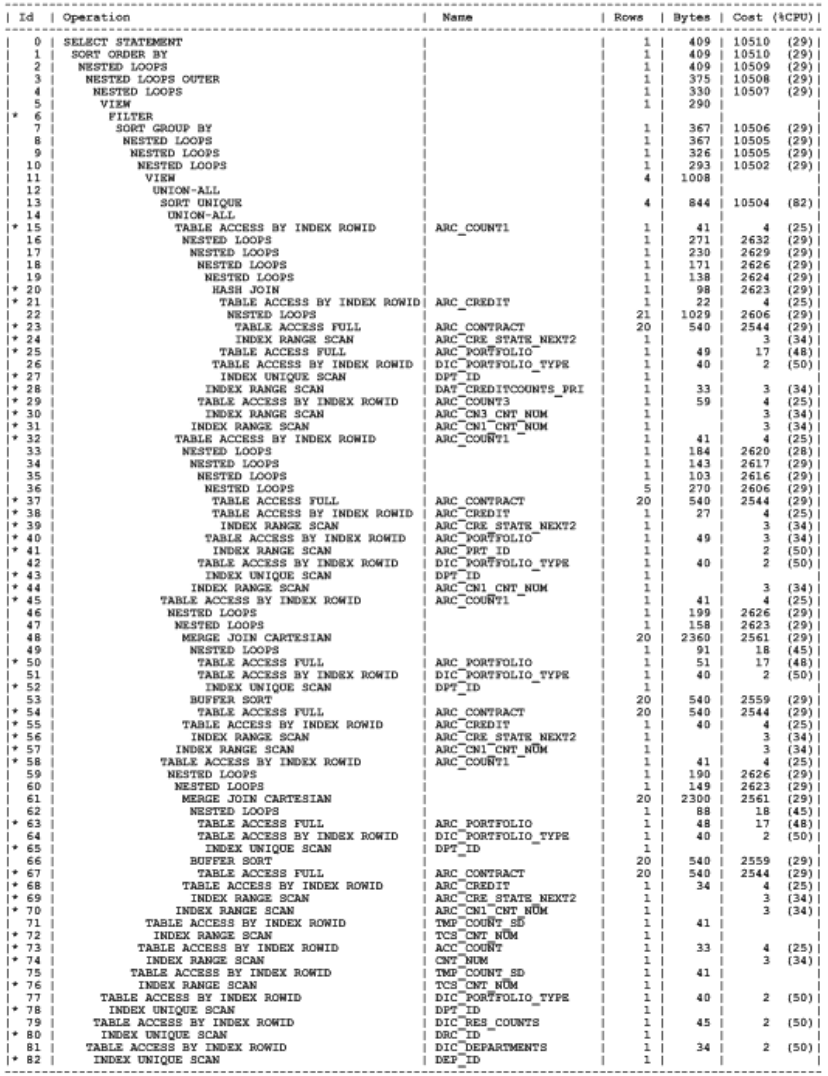
Метод экспертных оценок

Метод “серого” ящика



Настройка с участием администратора системы

Пример: Отчет “Кредитный портфель клиента”

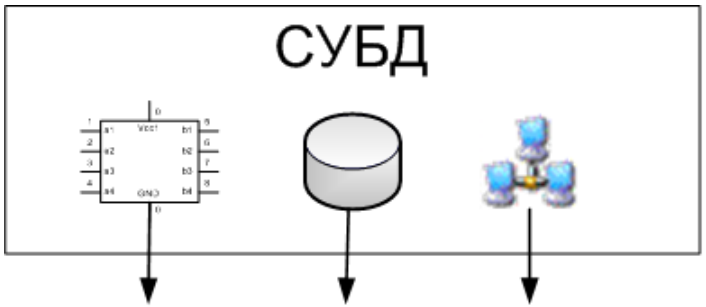


Id	Operation	Name	Rows	Bytes	Cost (%CPU)
0	SELECT STATEMENT		1	409	10510 (29)
1	SORT ORDER BY		1	409	10510 (29)
2	NESTED LOOPS		1	409	10509 (29)
3	NESTED LOOPS OUTER		1	375	10508 (29)
4	NESTED LOOPS		1	330	10507 (29)
5	VIEW		1	290	
* 6	FILTER				
7	SORT GROUP BY		1	367	10506 (29)
8	NESTED LOOPS		1	367	10505 (29)
9	NESTED LOOPS		1	326	10505 (29)
10	NESTED LOOPS		1	293	10502 (29)
11	VIEW		4	1008	
12	UNION-ALL				
13	SORT UNIQUE		4	844	10504 (82)
14	UNION-ALL				
* 15	TABLE ACCESS BY INDEX ROWID	ARC_COUNT1	1	41	4 (25)
16	NESTED LOOPS		1	271	2632 (29)
17	NESTED LOOPS		1	230	2629 (29)
18	NESTED LOOPS		1	171	2626 (29)
19	NESTED LOOPS		1	138	2624 (29)
* 20	HASH JOIN		1	98	2623 (29)
* 21	TABLE ACCESS BY INDEX ROWID	ARC_CREDIT	1	22	4 (25)
22	NESTED LOOPS		21	1029	2606 (29)
* 23	TABLE ACCESS FULL	ARC_CONTRACT	20	540	2544 (29)
* 24	INDEX RANGE SCAN	ARC_CRE_STATE_NEXT2	1		3 (34)
* 25	TABLE ACCESS FULL	ARC_PORTFOLIO	1	49	17 (48)
26	TABLE ACCESS BY INDEX ROWID	DIC_PORTFOLIO_TYPE	1	40	2 (50)
* 27	INDEX UNIQUE SCAN	DPT_ID	1		
* 28	INDEX RANGE SCAN	DAT_CREDITCOUNTS_PRI	1	33	3 (34)
* 29	TABLE ACCESS BY INDEX ROWID	ARC_COUNT3	1	59	4 (25)
* 30	INDEX RANGE SCAN	ARC_CN3_CNT_NUM	1		3 (34)
* 31	INDEX RANGE SCAN	ARC_CN1_CNT_NUM	1		3 (34)
* 32	TABLE ACCESS BY INDEX ROWID	ARC_COUNT1	1	41	4 (25)
33	NESTED LOOPS		1	184	2620 (28)
34	NESTED LOOPS		1	143	2617 (29)
35	NESTED LOOPS		1	103	2616 (29)
36	NESTED LOOPS		5	270	2606 (29)
* 37	TABLE ACCESS FULL	ARC_CONTRACT	20	540	2544 (29)
* 38	TABLE ACCESS BY INDEX ROWID	ARC_CREDIT	1	27	4 (25)
* 39	INDEX RANGE SCAN	ARC_CRE_STATE_NEXT2	1		3 (34)
* 40	TABLE ACCESS BY INDEX ROWID	ARC_PORTFOLIO	1	49	3 (34)
* 41	INDEX RANGE SCAN	ARC_PRT_ID	1		2 (50)
42	TABLE ACCESS BY INDEX ROWID	DIC_PORTFOLIO_TYPE	1	40	2 (50)
* 43	INDEX UNIQUE SCAN	DPT_ID	1		
* 44	INDEX RANGE SCAN	ARC_CN1_CNT_NUM	1		3 (34)
* 45	TABLE ACCESS BY INDEX ROWID	ARC_COUNT1	1	41	4 (25)
46	NESTED LOOPS		1	199	2626 (29)
47	NESTED LOOPS		1	158	2623 (29)
48	MERGE JOIN CARTESIAN		20	2360	2561 (29)
49	NESTED LOOPS		1	91	18 (45)
* 50	TABLE ACCESS FULL	ARC_PORTFOLIO	1	51	17 (48)
51	TABLE ACCESS BY INDEX ROWID	DIC_PORTFOLIO_TYPE	1	40	2 (50)
* 52	INDEX UNIQUE SCAN	DPT_ID	1		
53	BUFFER SORT		20	540	2559 (29)
* 54	TABLE ACCESS FULL	ARC_CONTRACT	20	540	2544 (29)
* 55	TABLE ACCESS BY INDEX ROWID	ARC_CREDIT	1	40	4 (25)
* 56	INDEX RANGE SCAN	ARC_CRE_STATE_NEXT2	1		3 (34)
* 57	INDEX RANGE SCAN	ARC_CN1_CNT_NUM	1		3 (34)
* 58	TABLE ACCESS BY INDEX ROWID	ARC_COUNT1	1	41	4 (25)
59	NESTED LOOPS		1	190	2626 (29)
60	NESTED LOOPS		1	149	2623 (29)
61	MERGE JOIN CARTESIAN		20	2300	2561 (29)
62	NESTED LOOPS		1	88	18 (45)
* 63	TABLE ACCESS FULL	ARC_PORTFOLIO	1	48	17 (48)
64	TABLE ACCESS BY INDEX ROWID	DIC_PORTFOLIO_TYPE	1	40	2 (50)
* 65	INDEX UNIQUE SCAN	DPT_ID	1		
66	BUFFER SORT		20	540	2559 (29)
* 67	TABLE ACCESS FULL	ARC_CONTRACT	20	540	2544 (29)
* 68	TABLE ACCESS BY INDEX ROWID	ARC_CREDIT	1	34	4 (25)
* 69	INDEX RANGE SCAN	ARC_CRE_STATE_NEXT2	1		3 (34)
* 70	INDEX RANGE SCAN	ARC_CN1_CNT_NUM	1		3 (34)
71	TABLE ACCESS BY INDEX ROWID	TMP_COUNT_SD	1	41	
* 72	INDEX RANGE SCAN	TCS_CNT_NUM	1		
* 73	TABLE ACCESS BY INDEX ROWID	ACC_COUNT	1	33	4 (25)
* 74	INDEX RANGE SCAN	CNT_NUM	1		3 (34)
75	TABLE ACCESS BY INDEX ROWID	TMP_COUNT_SD	1	41	
* 76	INDEX RANGE SCAN	TCS_CNT_NUM	1		
77	TABLE ACCESS BY INDEX ROWID	DIC_PORTFOLIO_TYPE	1	40	2 (50)
* 78	INDEX UNIQUE SCAN	DPT_ID	1		
79	TABLE ACCESS BY INDEX ROWID	DIC_RES_COUNTS	1	45	2 (50)
* 80	INDEX UNIQUE SCAN	DRC_ID	1		
81	TABLE ACCESS BY INDEX ROWID	DIC_DEPARTMENTS	1	34	2 (50)
* 82	INDEX UNIQUE SCAN	DEP_ID	1		

Объект	Количество строк	Размер, Мб
ACC_COUNT	440 227	1 317
ARC_CREDIT	407 961	1 030
ARC_CONTRACT	158 728	574
ARC_COUNT1	390 914	334
ARC_COUNT3	137 383	303
ARC_CN1_CNT_NUM	369 642	288
CNT_NUM	499 166	275
ARC_CRE_STATE_NEXT2	410 220	112
ARC_CN3_CNT_NUM	138 624	95
DAT_CREDITCOUNTS_P	84 505	58
ARC_PORTFOLIO	793	2
DIC_DEPARTMENTS	87	0
DIC_PORTFOLIO_TYPE	14	0
DPT_ID	14	0
TCS_CNT_NUM	0	0
DRC_ID	65	0
DEP_ID	87	0
ARC_PRT_ID	793	0
DIC_RES_COUNTS	65	0
TMP_COUNT_SD	0	0
Итого	3 039 288	4 388

Настройка с участием администратора системы

Объект и предмет исследования



Объект: процесс настройки системы баз данных

Предмет: стоимость выполнения запросов пользователей

Цель: сокращение сроков настройки системы баз данных на основе разработки метода анализа плана выполнения запроса, позволяющего определить стоимость выполнения запроса от параметров системы.

Новые результаты

1. Способ построения модели на основе свойств *O*-большое асимптотических оценок стоимости выполнения SQL запросов в зависимости от значимых параметров системы

2. Рекомендации по применению предложенной модели стоимости, учитывающие особенности предметной области и цели настройки.

3. Программный комплекс реализации системы тестирования запросов, используя предложенный метод анализа плана выполнения и гипотезу о стабильности поведения системы.

Условные обозначения

X – Стоимость

t – номер параметра модели (количество строк)

A – идентификатор левого операнда операции (например, чтение индекса)

B – идентификатор правого операнда операции (например, сканирование таблицы)

$w=F(w_A, w_B),\ w \in Q \setminus P,\ w_A, w_B \in Q$

$D(w_A)$ – вектор характеристик левого операнда-отношения w_A

$D(w_B)$ – вектор характеристик правого операнда-отношения w_B

$$D(w) = \begin{pmatrix} X \\ D_t(w) \end{pmatrix}$$

$$X = \Psi_w(D_t(w))$$

Способ построения модели стоимости

$$\Psi_w = O(g(D(w_A), D(w_B)))$$

$$\Psi_w \leq C \cdot g(D(w_A), D(w_B))$$

$$min_{C \in R^+} \sum_i \left(C \times g\left(D(w_A), D(w_B)\right) - X_i\right)^2$$

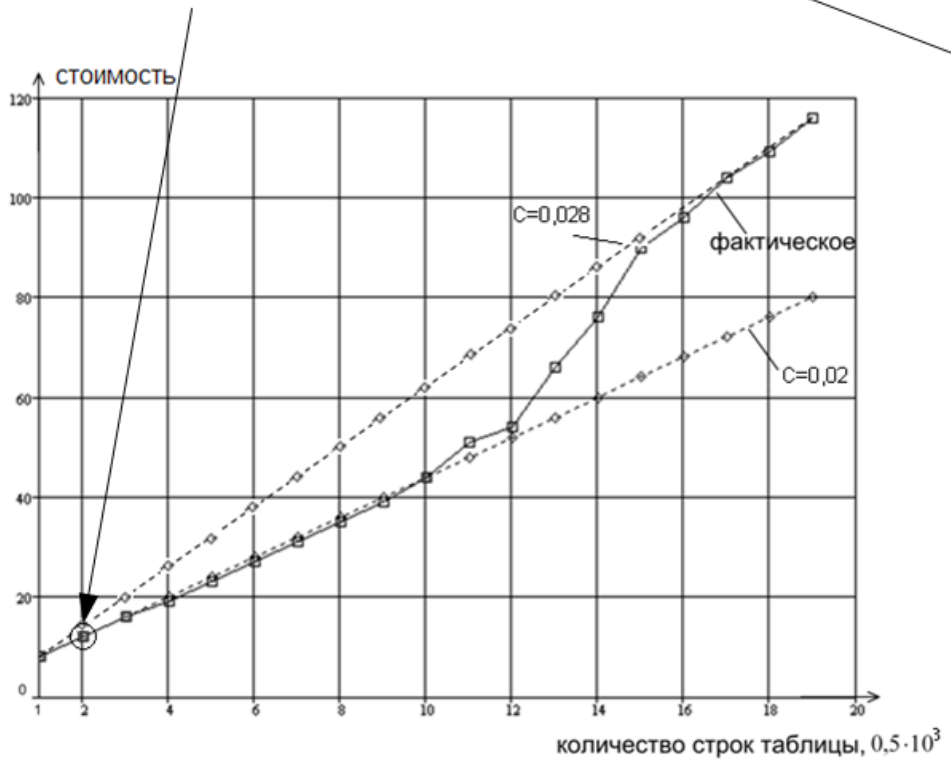
X_i – статистика выполнения запроса

Гипотеза о стабильности поведения системы: если математическая модель адекватна на границе области изменения параметров, то она адекватна всюду внутри области изменения параметров.

Пример: стоимость полного сканирования таблицы

```
SQL> SELECT * FROM hr.operative_data od;

Execution Plan
----------------------------------------------------------------------------------
| Id  | Operation          | Name           | Rows  | Cost (%CPU)| Time     |
----------------------------------------------------------------------------------
|   0 |   TABLE ACCESS FULL| OPERATIVE_DATA |  1000 |    20     (1)| 00:00:03 |
----------------------------------------------------------------------------------
```



$$g = n_R$$

$$\Psi_w = C \cdot n_R$$

$$C = \frac{20}{1000}$$

$$D(w) = \begin{pmatrix} 0,02n_R \\ n_R \end{pmatrix}$$

Пример: соединение вложенными циклами

Алгоритм соединения блоками с использованием вложенных циклов

```
// Дано:    Отношение внешнего цикла R
//          Количество кортежей N_R(R) отношения R
//          Отношение внутреннего цикла S
//          Количество кортежей N_S|Ri(S) отношения S, участвующие в
//          операции при условии, что выбран кортеж R_i отношения R
//          Процедура чтения кортежа read_tuple

For R_i=1 to N_R(R) {                          | N_R
Rtuple = read_tuple(R,R_i)                     | A
        For S_i=1 to N_S|Ri(R) {               | N_S
        Stuple = read_tuple(S,S_i)             | B
        If Rtuple == Stuple then               | D
            <поместить кортеж                  | E
            в новое отношение>
        }
}
```

$$\Psi_{w,} = N_R\left(A + N_S(B + D + E)\right)$$

$$\Psi_{w,} = O(N_R \times N_S)$$

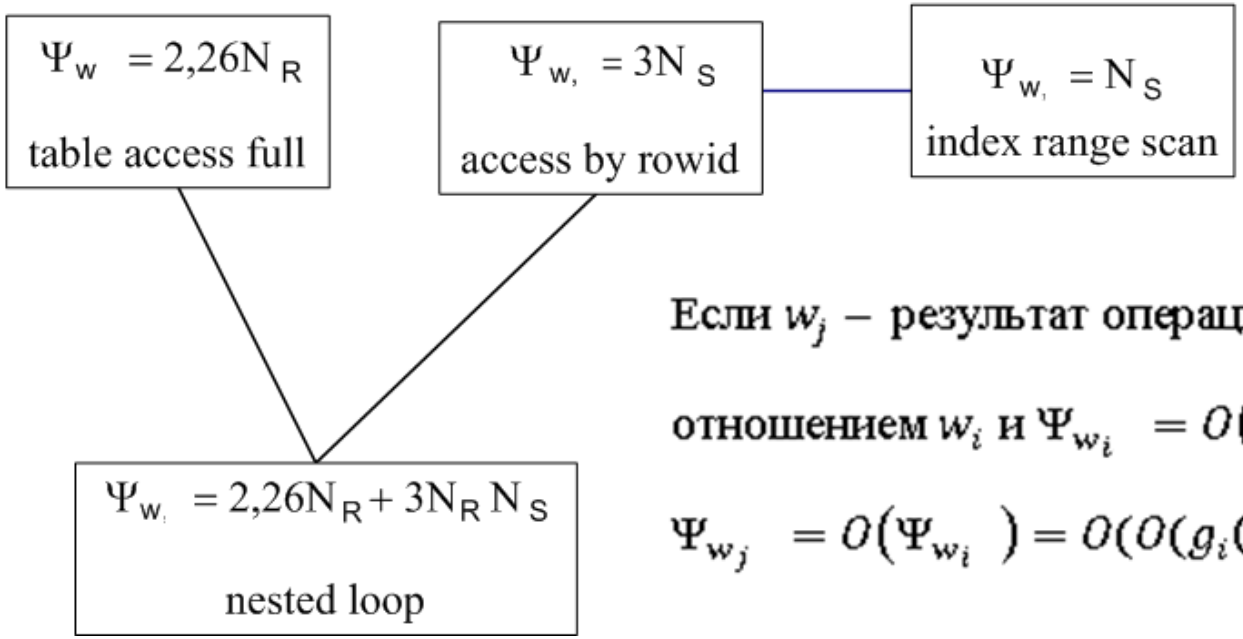
$$\exists C: \Psi_{w,} \leq C \times N_R N_S$$

Известно значение $\Psi_{w,}$ для текущего значения N_R и N_S

$$C = \frac{\Psi_w}{N_R \times N_S}$$

Пример: соединение вложенными циклами

Стоимость	Строк	Операция
526	113	Select statement
526	113	nested loops
226	100	table access full
3	1	table access by index rowid
1	1	index range scan



Если w_j – результат операции над производным отношением w_i и $\Psi_{w_i} = O(g_i(n_i))$, то

$$\Psi_{w_j} = O(\Psi_{w_i}) = O(O(g_i(n_i))) = O(g_i(n_i)).$$

Пример: решение 1

```
SELECT t_page.page_id, t_page.content
FROM t_page
    , (SELECT COUNT(kw_id) AS cnt, page_id
       FROM  t_kw_on_page
       WHERE kw_id IN (1,2,3)
       GROUP BY page_id)  T1
WHERE T1.cnt = 3
  AND t_page.page_id = T1.page_id;
```

$$\Psi = 2 + 1{,}462 \times n \times 10^{-5} + 7400 \times m \times 10^{-5}$$

```
Execution Plan
----------------------------------------------------------
Plan hash value: 1061824386

-------------------------------------------------------------------------------------------------
| Id  | Operation             | Name             | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                  |    10 |   240 |     19  (11)| 00:00:01 |
|*  1 |  HASH JOIN            |                  |    10 |   240 |     19  (11)| 00:00:01 |
|   2 |   VIEW                |                  |    10 |   130 |     15   (7)| 00:00:01 |
|*  3 |    FILTER             |                  |       |       |             |          |
|   4 |     HASH GROUP BY     |                  |    10 |    60 |     15   (7)| 00:00:01 |
|   5 |      INLIST ITERATOR  |                  |       |       |             |          |
|*  6 |       INDEX RANGE SCAN| T_KW_ON_PAGE_PK  |  2997 | 17982 |     14   (0)| 00:00:01 |
|   7 |   TABLE ACCESS FULL   | T_PAGE           |   999 | 10989 |      3   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   1 - access("T_PAGE"."PAGE_ID"="T1"."PAGE_ID")
   3 - filter(COUNT(*)=3)
   6 - access("KW_ID"=1 OR "KW_ID"=2 OR "KW_ID"=3)
```

Пример: решение 2

```
SELECT * FROM t_page
WHERE page_id IN
   (SELECT page_id
    FROM (SELECT COUNT(kw_id) AS c,
                 page_id
          FROM t_kw_on_page
          WHERE kw_id IN
              (SELECT kw_id
               FROM t_keyword
               WHERE keyword='tag 1')
             OR kw_id IN
             (SELECT kw_id
              FROM t_keyword
              WHERE keyword='tag 2')
             OR kw_id IN
             (SELECT kw_id
              FROM t_keyword
              WHERE keyword='tag 3')
          GROUP BY page_id)
    WHERE c > 2);
```

```
Execution Plan
----------------------------------------------------------------------------------------------------
| Id  | Operation                      | Name          | Rows  | Bytes | Cost (%CPU)| Time     |
----------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT               |               |    50 |  1200 |   501  (17)| 00:00:07 |
|*  1 |  HASH JOIN RIGHT SEMI          |               |    50 |  1200 |   501  (17)| 00:00:07 |
|   2 |   VIEW                         | VW_NSO_1      |    50 |   650 |   498  (18)| 00:00:06 |
|*  3 |    FILTER                      |               |       |       |            |          |
|   4 |     HASH GROUP BY              |               |    50 |   300 |   498  (18)| 00:00:06 |
|*  5 |      FILTER                    |               |       |       |            |          |
|   6 |       TABLE ACCESS FULL        | T_KW_ON_PAGE  |  998K|  5847K|   431   (5)| 00:00:06 |
|*  7 |       TABLE ACCESS BY INDEX ROWID | T_KEYWORD  |     1 |    10 |     2   (0)| 00:00:01 |
|*  8 |        INDEX UNIQUE SCAN       | SYS_C0028821  |     1 |       |     1   (0)| 00:00:01 |
|*  9 |        TABLE ACCESS BY INDEX ROWID | T_KEYWORD |     1 |    10 |     2   (0)| 00:00:01 |
|* 10 |         INDEX UNIQUE SCAN      | SYS_C0028821  |     1 |       |     1   (0)| 00:00:01 |
|* 11 |         TABLE ACCESS BY INDEX ROWID| T_KEYWORD |     1 |    10 |     2   (0)| 00:00:01 |
|* 12 |          INDEX UNIQUE SCAN     | SYS_C0028821  |     1 |       |     1   (0)| 00:00:01 |
|  13 |   TABLE ACCESS FULL            | T_PAGE        |   999 | 10989 |     3   (0)| 00:00:01 |
----------------------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - access("PAGE_ID"="$nso_col_1")
   3 - filter(COUNT(*)>2)
   5 - filter( EXISTS (SELECT 0 FROM "T_KEYWORD" "T_KEYWORD" WHERE "KW_ID"=:B1 AND
              "KEYWORD"='tag 1') OR  EXISTS (SELECT 0 FROM "T_KEYWORD" "T_KEYWORD" WHERE "KW_ID"=:B2 AND
              "KEYWORD"='tag 2') OR  EXISTS (SELECT 0 FROM "T_KEYWORD" "T_KEYWORD" WHERE "KW_ID"=:B3 AND
              "KEYWORD"='tag 3'))
   7 - filter("KEYWORD"='tag 1')
   8 - access("KW_ID"=:B1)
   9 - filter("KEYWORD"='tag 2')
  10 - access("KW_ID"=:B1)
  11 - filter("KEYWORD"='tag 3')
  12 - access("KW_ID"=:B1)
```

$$\Psi = 49{,}94 \times n \times 10^{-5} + 300 \times m \times 10^{-5}$$

Пример: анализ на основе значимого показателя

Решение 1

```
SELECT t_page.page_id, t_page.content
FROM t_page
   , (SELECT COUNT(kw_id) AS cnt, page_id
      FROM  t_kw_on_page
      WHERE kw_id IN (1,2,3)
      GROUP BY page_id)  T1
WHERE T1.cnt = 3
  AND t_page.page_id = T1.page_id;
```

Решение 2

```
SELECT * FROM t_page
WHERE page_id IN
   (SELECT page_id
    FROM (SELECT COUNT(kw_id) AS c,
                 page_id
          FROM t_kw_on_page
          WHERE kw_id IN
             (SELECT kw_id
              FROM t_keyword
              WHERE keyword='tag 1')
            OR kw_id IN
             (SELECT kw_id
              FROM t_keyword
              WHERE keyword='tag 2')
            OR kw_id IN
             (SELECT kw_id
              FROM t_keyword
              WHERE keyword='tag 3')
          GROUP BY page_id)
    WHERE c > 2);
```

$$\Psi_{\text{решение 1}} = 16 + 7400 \times m \times 10^{-5}$$

$$\Psi_{\text{решение 2}} = 3 + 49{,}94 \times n \times 10^{-5}$$

Номер решения	Модель стоимости	Модель по критерию значимости показателя	Значимый показатель	Коэффициент при показателе
1	$2 + (1{,}462n + 7400m)*10^{-5}$	r:90 $n:76+1{,}462n*10^{-5}$ $m:16 + 7400m*10^{-5}$	m	$7400*10^{-5}$
2	$(49{,}94n + 300m)*10^{-5}$	r:501 $n:3+49{,}94n*10^{-5}$ $m:498 + 300m*10^{-5}$	n	$49{,}94*10^{-5}$

Настройка с участием администратора системы

Пример: анализ на основе параметрических моделей

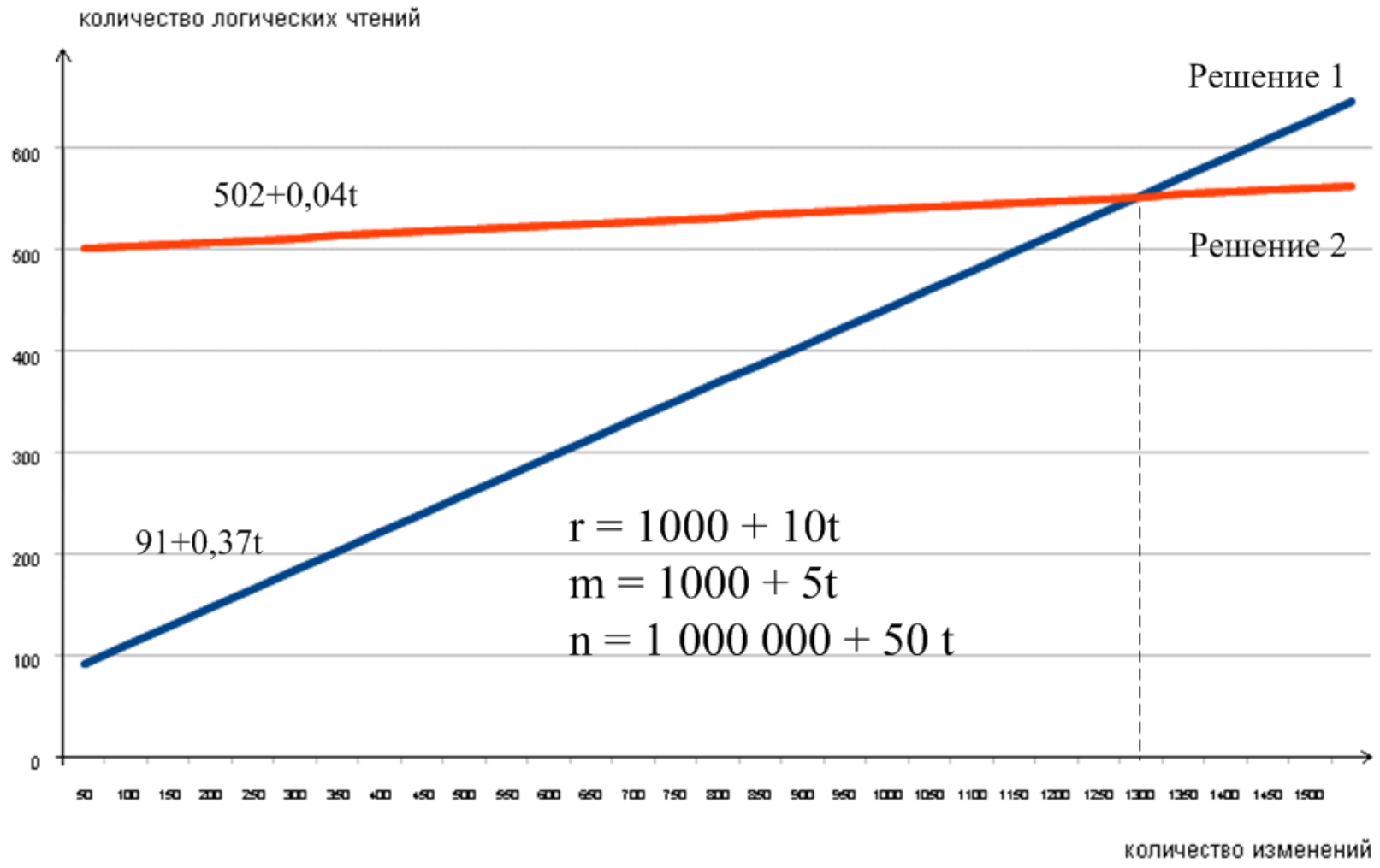
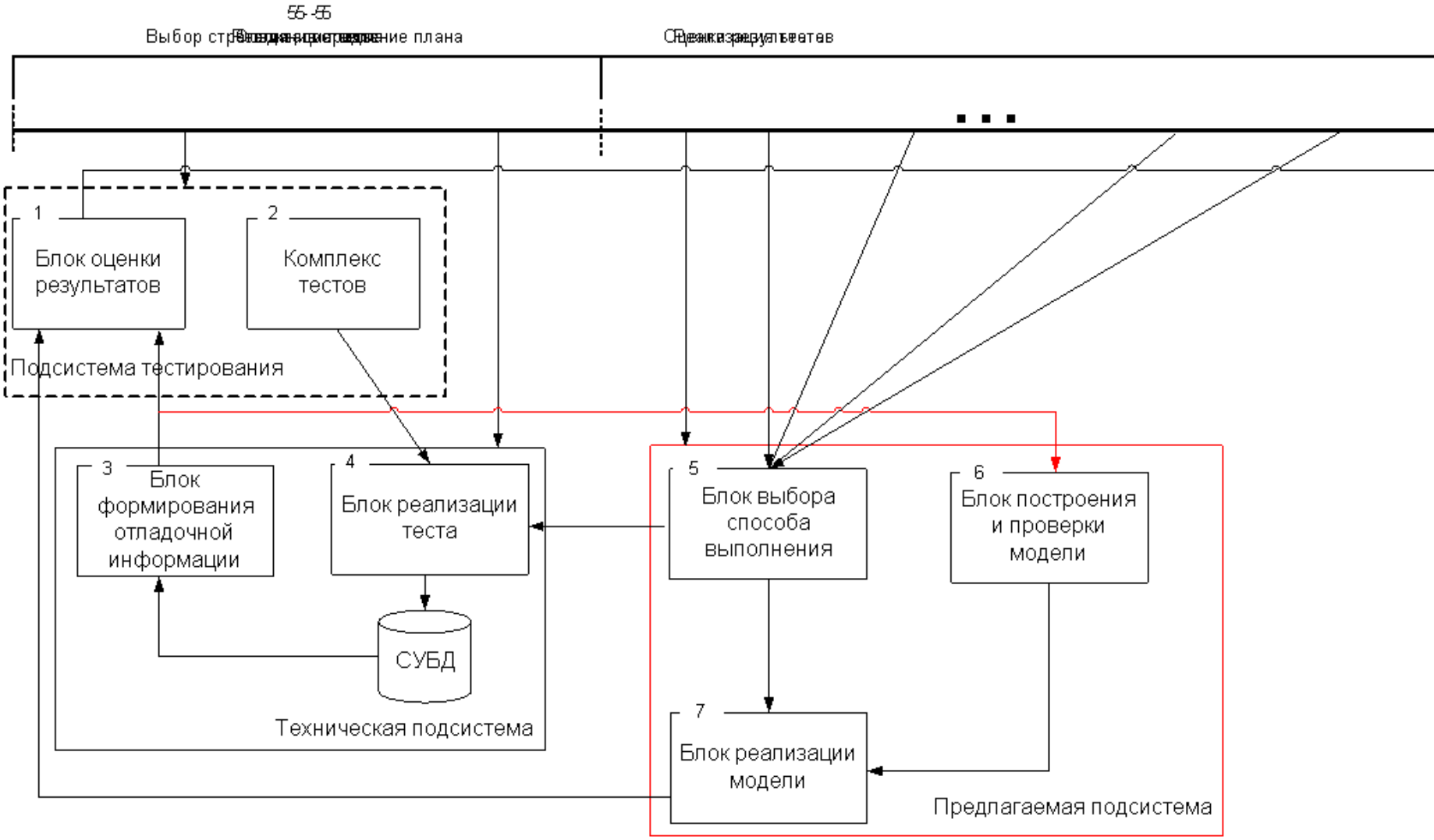


Схема нагрузочного тестирования



Настройка с участием администратора системы

Пример. Отчет “кредитный портфель клиента”

Параметры:

n_{AC} – количество договоров, по которым возможно предоставление кредита;

r_{AC_PRI} – количество связанных с договором обязательств;

n_{AP} – количество типов договоров;

r_{CN1_NUM} – количество связанных с договором внутрибанковских счетов;

r_{CN3_NUM} – количество связанных с договором пассивных счетов;

r_{CNT_NUM} – количество контрагентов по договору.

Модели затрат ресурсов, определенные по критерию значимости статистического показателя:

Показатель	Модель
n_{AC}	$0{,}39 + 6{,}06n_{AC}$
r_{AC_PRI}	$80996 + 32169r_{AC\_PRI}$
n_{AP}	$658957 + 76{,}46n_{AP}$
r_{CN1_NUM}	$637773 + 10974r_{CN1\_NUM}$
r_{CN3_NUM}	$641694 + 9014r_{CN3\_NUM}$
r_{CNT_NUM}	$644742 + 14980r_{CNT\_NUM}$

Модель запроса:

$$\Psi_{\text{общая}} = (60{,}64n_{AC} + 13{,}76n_{AC}r_{CNT\_NUM} + 27{,}41n_{AC}r_{AC\_PRI} + 3{,}9n_{AP} + 0{,}39 \times 10^{-2}n_{AC}r_{AC\_PRI}n_{AP} + 0{,}56n_{AC}r_{AC\_PRI}r_{CN1\_NUM} + 0{,}46n_{AC}r_{AC\_PRI}r_{CN3\_NUM}) \times 10^{-2}$$

Пример. Отчет “кредитный портфель клиента”

Рекомендации:

```
WITH credit_view AS

     ( SELECT curr_credit.*, prt_id, prt_name, prt_dpr_id, prt_dep_id,
prt_cur_typ, prt_dpt_id, prt_risk_rate, prt_prcbad_res_cnt,
prt_prcbad_res_vir, prt_prc_res_vir, prt_prc_res_cnt, prt_tax_res_cnt,
prt_tax_res_vir, prt_fine_res_cnt, prt_fine_res_vir
         FROM curr_portfolio, curr_credit, acc_contract
        WHERE prt_deleted = 0
          AND cre_prt_id = prt_id
          AND con_id = cre_con_id
          AND con_dt_end > :p_buh_date)
```

Модель запроса:

$$\Psi_{\text{общая}} = (25{,}77n_{AC} + 54{,}6n_{AP} + 0{,}947n_{CRC} + 6{,}21n_{CN3} + 34{,}66n_{AC}r_{CN1\_NUM} + 32{,}96n_{AC}r_{CNT\_NUM}) \times 10^{-2}$$

Модели затрат ресурсов, определенные по критерию значимости статистического показателя

Показатель	Модель
n_{AC}	$4587 + 1{,}28n_{AC}$
n_{CRC}	$143349 + 0{,}00947n_{CRC}$
n_{AP}	$144101 + 54{,}6n_{AP}$
r_{CN1_NUM}	$68602 + 37780r_{CN1\_NUM}$
n_{CN3}	$140283 + 0{,}0621n_{CN3}$
r_{CNT_NUM}	$108235 + 35926r_{CNT\_NUM}$

Модели затрат ресурсов, определенные по критерию значимости производного отношения

Операция	Модель	Результат применения оператора L	Позиция по критерию значимости
NESTED LOOPS	$32{,}96n_{AC}r_{CNT\_NUM} \times 10^{-2}$	$32{,}96n_{AC} \times 10^{-2}$	1
TABLE ACCESS BY INDEX ROWID OBJ#(23439)	$(0{,}542n_{AC} + 0{,}947n_{CRC} + 6{,}21n_{CN3} + 20{,}49n_{AC}r_{CN1\_NUM}) \times 10^{-2}$	$22{,}6n_{AC} \times 10^{-2}$	2
NESTED LOOPS	$(0{,}542n_{AC} + 0{,}947n_{CRC} + 6{,}21n_{CN3} + 9{,}19n_{AC}r_{CN1\_NUM}) \times 10^{-2}$	$18{,}38n_{AC} \times 10^{-2}$	3
NESTED LOOPS	$(0{,}542n_{AC} + 5{,}14n_{AC}r_{CN1\_NUM}) \times 10^{-2}$	$10{,}28n_{AC} \times 10^{-2}$	4
TABLE ACCESS BY INDEX ROWID OBJ#(23439)	$(0{,}542n_{AC} + 8{,}04n_{AC}r_{CN1\_NUM}) \times 10^{-2}$	$5{,}8n_{AC} \times 10^{-2}$	5
NESTED LOOPS	$(0{,}542n_{AC} + 2{,}38n_{AC}r_{CN1\_NUM}) \times 10^{-2}$	$5{,}76n_{AC} \times 10^{-2}$	6
TABLE ACCESS BY INDEX ROWID OBJ#(23439)	$(0{,}542n_{AC} + 3{,}56n_{AC}r_{CN1\_NUM}) \times 10^{-2}$	$2{,}34n_{AC} \times 10^{-2}$	7
TABLE ACCESS BY INDEX ROWID OBJ#(23439)	$(0{,}542n_{AC} + 2{,}57n_{AC}r_{CN1\_NUM}) \times 10^{-2}$	$2{,}58n_{AC} \times 10^{-2}$	8

Модель запроса:

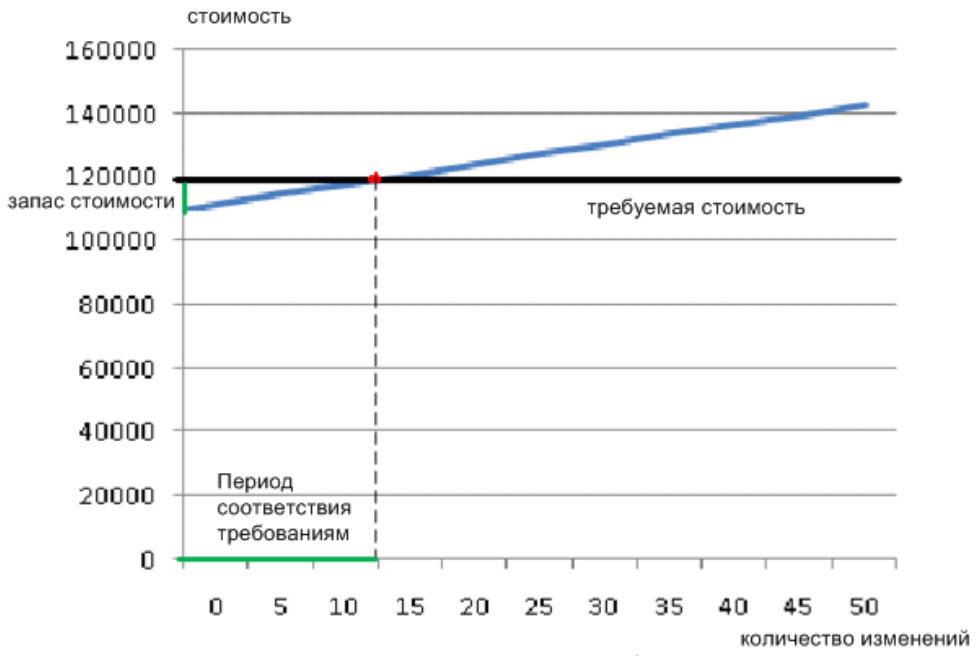
$$\Psi_{\text{общая}} = (25{,}77n_{AC} + 54{,}6n_{AP} + 0{,}947n_{CRC} + 6{,}21n_{CN3} + 34{,}66n_{AC}r_{CN1\_NUM} + 0{,}858n_{CNT\_NUM}) \times 10^{-2},$$

Настройка с участием администратора системы

Пример. Отчет “кредитный портфель клиента”

Размер тестовых данных

Название таблицы	Значение параметра
curr_credit	$n_{AC} = 108865$
curr_credit1	$n_{AC} = 217730$
. . .	. . .



$$\Psi_{\text{общая}}(t) = 111811 + 621 \times t,$$

$$\Psi_{\text{общая}}(t) < 120\,000.$$

Настройка с участием администратора системы

Основные результаты

1. Для изучения стоимости на основе свойств асимптотических оценок предложено использовать гипотезу стабильности поведения системы.

2. В интересах сокращения количества учитываемых параметров разработан способ построения модели стоимости на основе свойств *O*-большое асимптотических оценок, позволяющий перейти от рекурсивного способа вычисления стоимости к замкнутому.

3.На основе разработанной модели стоимости определены:

3.1. показатели значимых параметров, позволяющие производить ранжирование операций или параметров и тем самым сократить переборные схемы анализа;
3.2. параметрические модели, позволяющие экстраполировать результаты тестовых испытаний на основе имеющихся данных.