

Среда реализации MySQL 8.0.3, MySQL Workbench 8.0.

## Создание структуры данных

Для создания БД воспользуемся скриптом

```
CREATE DATABASE IF NOT EXISTS mydb;
USE mydb;
```

Создаем таблицы с внешними ключами:

```
CREATE TABLE users (
    id INT(11) UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    password VARCHAR(50) NOT NULL,
    email VARCHAR(50) NOT NULL,
    INDEX (username),
    INDEX (password),
    INDEX (email)
) ENGINE=InnoDB;
```

```
CREATE TABLE posts (
    id INT(11) UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    user_id INT(11) UNSIGNED NOT NULL,
    title VARCHAR(100) NOT NULL,
    content TEXT NOT NULL,
    created_at DATETIME NOT NULL,
    INDEX (user_id),
    INDEX (title),
    INDEX (content),
    INDEX (created_at)
) ENGINE=InnoDB;
```

```
CREATE TABLE comments (
    id INT(11) UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    post_id INT(11) UNSIGNED NOT NULL,
    user_id INT(11) UNSIGNED NOT NULL,
    comment TEXT NOT NULL,
    created_at DATETIME NOT NULL,
    INDEX (post_id),
    INDEX (user_id),
    INDEX (comment),
    INDEX (created_at)
) ENGINE=InnoDB;
```

```
CREATE TABLE tags (
    id INT(11) UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    tag VARCHAR(50) NOT NULL,
    INDEX (tag)
) ENGINE=InnoDB;
```

```
CREATE TABLE tag_posts (
    tag_id INT(11) UNSIGNED NOT NULL,
    post_id INT(11) UNSIGNED NOT NULL,
    PRIMARY KEY (tag_id, post_id),
    INDEX (tag_id),
    INDEX (post_id)
) ENGINE=InnoDB;
```

```
CREATE TABLE tag_comments (
    tag_id INT(11) UNSIGNED NOT NULL,
    comment_id INT(11) UNSIGNED NOT NULL,
    PRIMARY KEY (tag_id, comment_id),
    INDEX (tag_id),
    INDEX (comment_id)
) ENGINE=InnoDB;
```

```
CREATE TABLE tag_users (
    tag_id INT(11) UNSIGNED NOT NULL,
    user_id INT(11) UNSIGNED NOT NULL,
    PRIMARY KEY (tag_id, user_id),
    INDEX (tag_id),
    INDEX (user_id)
) ENGINE=InnoDB;
```

```
CREATE TABLE tag_posts_users (
    tag_id INT(11) UNSIGNED NOT NULL,
    post_id INT(11) UNSIGNED NOT NULL,
    user_id INT(11) UNSIGNED NOT NULL,
    PRIMARY KEY (tag_id, post_id, user_id),
    INDEX (tag_id),
    INDEX (post_id),
    INDEX (user_id)
) ENGINE=InnoDB;
```

```
CREATE TABLE tag_comments_users (
    tag_id INT(11) UNSIGNED NOT NULL,
    comment_id INT(11) UNSIGNED NOT NULL,
    user_id INT(11) UNSIGNED NOT NULL,
    PRIMARY KEY (tag_id, comment_id, user_id),
    INDEX (tag_id),
    INDEX (comment_id),
    INDEX (user_id)
) ENGINE=InnoDB;
```

The figure consists of two 3x3 grids of dots. The left grid represents the initial state, and the right grid represents the final state. In the initial state, the top-left 2x2 block of dots is black, while the rest are white. In the final state, the top-right 2x2 block of dots is black, while the rest are white.

```

#####
#      #
#      #
#####
####  ####  #  #####  #

```

[illegible]

Figure 1 shows two 3x3 grids of dots. The left grid represents the initial state, and the right grid represents the final state. The dots are arranged in a 3x3 pattern, with the left grid showing a single dot in each cell and the right grid showing a single dot in each cell.

1  
 2  
 3  
 4  
 5  
 6  
 7  
 8  
 9  
 10  
 11  
 12  
 13  
 14  
 15  
 16  
 17  
 18  
 19  
 20  
 21  
 22  
 23  
 24  
 25  
 26  
 27  
 28  
 29  
 30  
 31  
 32  
 33  
 34  
 35  
 36  
 37  
 38  
 39  
 40  
 41  
 42  
 43  
 44  
 45  
 46  
 47  
 48  
 49  
 50  
 51  
 52  
 53  
 54  
 55  
 56  
 57  
 58  
 59  
 60  
 61  
 62  
 63  
 64  
 65  
 66  
 67  
 68  
 69  
 70  
 71  
 72  
 73  
 74  
 75  
 76  
 77  
 78  
 79  
 80  
 81  
 82  
 83  
 84  
 85  
 86  
 87  
 88  
 89  
 90  
 91  
 92  
 93  
 94  
 95  
 96  
 97  
 98  
 99  
 100  
 101  
 102  
 103  
 104  
 105  
 106  
 107  
 108  
 109  
 110  
 111  
 112  
 113  
 114  
 115  
 116  
 117  
 118  
 119  
 120  
 121  
 122  
 123  
 124  
 125  
 126  
 127  
 128  
 129  
 130  
 131  
 132  
 133  
 134  
 135  
 136  
 137  
 138  
 139  
 140  
 141  
 142  
 143  
 144  
 145  
 146  
 147  
 148  
 149  
 150  
 151  
 152  
 153  
 154  
 155  
 156  
 157  
 158  
 159  
 160  
 161  
 162  
 163  
 164  
 165  
 166  
 167  
 168  
 169  
 170  
 171  
 172  
 173  
 174  
 175  
 176  
 177  
 178  
 179  
 180  
 181  
 182  
 183  
 184  
 185  
 186  
 187  
 188  
 189  
 190  
 191  
 192  
 193  
 194  
 195  
 196  
 197  
 198  
 199  
 200  
 201  
 202  
 203  
 204  
 205  
 206  
 207  
 208  
 209  
 210  
 211  
 212  
 213  
 214  
 215  
 216  
 217  
 218  
 219  
 220  
 221  
 222  
 223  
 224  
 225  
 226  
 227  
 228  
 229  
 230  
 231  
 232  
 233  
 234  
 235  
 236  
 237  
 238  
 239  
 240  
 241  
 242  
 243  
 244  
 245  
 246  
 247  
 248  
 249  
 250  
 251  
 252  
 253  
 254  
 255  
 256  
 257  
 258  
 259  
 260  
 261  
 262  
 263  
 264  
 265  
 266  
 267  
 268  
 269  
 270  
 271  
 272  
 273  
 274  
 275  
 276  
 277  
 278  
 279  
 280  
 281  
 282  
 283  
 284  
 285  
 286  
 287  
 288  
 289  
 290  
 291  
 292  
 293  
 294  
 295  
 296  
 297  
 298  
 299  
 300  
 301  
 302  
 303  
 304  
 305  
 306  
 307  
 308  
 309  
 310  
 311  
 312  
 313  
 314  
 315  
 316  
 317  
 318  
 319  
 320  
 321  
 322  
 323  
 324  
 325  
 326  
 327  
 328  
 329  
 330  
 331  
 332  
 333  
 334  
 335  
 336  
 337  
 338  
 339  
 340  
 341  
 342  
 343  
 344  
 345  
 346  
 347  
 348  
 349  
 350  
 351  
 352  
 353  
 354  
 355  
 356  
 357  
 358  
 359  
 360  
 361  
 362  
 363  
 364  
 365  
 366  
 367  
 368  
 369  
 370  
 371  
 372  
 373  
 374  
 375  
 376  
 377  
 378  
 379  
 380  
 381  
 382  
 383  
 384  
 385  
 386  
 387  
 388  
 389  
 390  
 391  
 392  
 393  
 394  
 395  
 396  
 397  
 398  
 399  
 400  
 401  
 402  
 403  
 404  
 405  
 406  
 407  
 408  
 409  
 410  
 411  
 412  
 413  
 414  
 415  
 416  
 417  
 418  
 419  
 420  
 421  
 422  
 423  
 424  
 425  
 426  
 427  
 428  
 429  
 430  
 431  
 432  
 433  
 434  
 435  
 436  
 437  
 438  
 439  
 440  
 441  
 442  
 443  
 444  
 445  
 446  
 447  
 448  
 449  
 450  
 451  
 452  
 453  
 454  
 455  
 456  
 457  
 458  
 459  
 460  
 461  
 462  
 463  
 464  
 465  
 466  
 467  
 468  
 469  
 470  
 471  
 472  
 473  
 474  
 475  
 476  
 477  
 478  
 479  
 480  
 481  
 482  
 483  
 484  
 485  
 486  
 487  
 488  
 489  
 490  
 491  
 492  
 493  
 494  
 495  
 496  
 497  
 498  
 499  
 500  
 501  
 502  
 503  
 504  
 505  
 506  
 507  
 508  
 509  
 510  
 511  
 512  
 513  
 514  
 515  
 516  
 517  
 518  
 519  
 520  
 521  
 522  
 523  
 524  
 525

Figure 1 shows two 3x3 grids representing the 8-puzzle state. The left grid is the initial state, and the right grid is the final state after a sequence of moves. The tiles are numbered 1 through 8, and the blank space is represented by 0.

1	2	3
4	5	6
7	8	0

1	2	3
4	5	6
7	8	0

[illegible][illegible]

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100

Figure 1 shows two 3x3 grids of dots. The left grid represents the initial state, and the right grid represents the final state after a sequence of moves. The dots are arranged as follows:

Initial State	Final State
•   •   •	•   •   •
•   •   •	•   •   •
•   •   •	•   •   •

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100  
 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100  
 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

The figure consists of two 3x3 grids of dots. The left grid represents the initial state, with dots at positions (1,1), (1,3), (2,2), (3,1), and (3,3). The right grid represents the final state, with dots at positions (1,2), (1,3), (2,1), (2,2), (2,3), (3,1), (3,2), and (3,3). Coordinates are given as (row, column) starting from (1,1) at the top-left.

[illegible]

Заполняем таблицы данными используя стандартную конструкцию

Figure 1. The 1000 Genomes Project. The figure is divided into two main sections: 'Genomes' and 'Variants'. The 'Genomes' section shows a grid of 1000 human genomes, with a small inset showing a single genome. The 'Variants' section shows a grid of variants across the 1000 genomes, with a small inset showing a single variant. The figure is a schematic representation of the 1000 Genomes Project, showing the distribution of genetic variation across a diverse population.

## Выполнение заданий

### Задание 1.

Найти информацию о всех контрактах, связанных с сотрудниками департамента «Logistic». Вывести: contract\_id, employee\_name.

Запрос

```

SELECT contract_id, employee_name
FROM employees
WHERE department = 'Logistic'

```

employee_name	contract_id
Egor Egorov	3
Egor Egorov	8
Egor Egorov	18
Egor Egorov	23
Alex Alexeev	20

## Задание 2

Найти среднюю стоимость контрактов, заключенных сотрудников Ivan Ivanov. Вывести: среднее значение amount

Запрос:

```

SELECT AVG(amount)
FROM employees
WHERE employee_name = 'Ivan Ivanov'

```

□ □ □ □ □ □ □ □ □ □

100%	43:1
Result Grid	
amount	
40000.0000	

## Задание 3

Найти самую часто встречающуюся локацию среди всех заказчиков. Вывести: location, count

Запрос:

```

SELECT location, COUNT(*)
FROM orders
GROUP BY location
ORDER BY COUNT(*) DESC

```

□ □

	location	count	
►	Moscow	3	

#### Задание 4

Найти контракты одинаковой стоимости. Вывести count, amount

Запрос:

```
SELECT count, amount
FROM contracts
GROUP BY amount
```

:

count	amount
2	30000
2	50000
2	60000
2	150000

#### Задание 5

Найти заказчика с наименьшей средней стоимостью контрактов. Вывести customer\_name, среднее значение amount

Запрос:

```
SELECT customer_name, avg(amount)
```

```
FROM contracts
GROUP BY customer_name
ORDER BY avg(amount) ASC
LIMIT 1
```

```
SELECT customer_name, avg(amount)
```

```
FROM contracts
GROUP BY customer_name
ORDER BY avg(amount) ASC
LIMIT 1
```

:

customer_name	Avarage
► Andrew Nilov	48666.6667

#### Задание 6

Найти отдел, заключивший контрактов на наибольшую сумму. Вывести:  
department\_name, sum

Запрос:

```
SELECT department_name, SUM(amount) AS sum
FROM contracts
GROUP BY department_name
ORDER BY sum DESC
LIMIT 1
```

:

Result Grid	
department_name	sum
Economy	610000