## Итоговый лист

|  |  |  |
| --- | --- | --- |
| День | Номер задания | Ваш ответ |
| 1 | 2 | Table 1. № person(primary key) and Name(alternate key) = composite key.  Table 2 Standing - potential key  table 3 id - primary key(simple)  Table 4 Grade - primary key  table1.Specialty = table3.id  table1.grade = table4.grade  between table 1 and table 2 it may not be possible to establish a direct connection between these tables because data in Standing for table 1 is String, but in table 2 in Range of values. In such cases, we might need to consider alternative approaches, such as creating an intermediary table or modifying the data structure to establish a connection. |
| 2 | 5 | CREATE TABLE MyCourseTable (  ID INT NOT NULL AUTO\_INCREMENT,  Lname VARCHAR(255) NOT NULL,  Fname VARCHAR(255) NOT NULL,  Gender ENUM('M', 'F') NOT NULL,  Specialty VARCHAR(255) NOT NULL,  Grade VARCHAR(255) NOT NULL,  Start\_date DATE NOT NULL,  PRIMARY KEY (ID)  ); |
|  |  | INSERT INTO MyCourseTable (Lname, Fname, Gender, Specialty, Grade, Start\_date)  VALUES('Ivanova','Katy','F','1','Middle','2015-10-22'),  ('Sidorov','Alex','M','3','Senior','2015-03-01'),  ('Kruticov','Edvard','M','2','Middle','2016-01-10'),  ('Krukova','Nina','F','4','Junior','2014-05-30'),  ('Lavrova','Dina','F','3','Middle','2015-12-30'),  ('Stepanenko','Ann','F','3','Intern',NULL),  ('Lavrov','Gosha','M','1','Senior','2015-12-05'),  ('Kolchin', 'Gena', 'M','1','Middle','2016-01-01'),  ('Kushina','Lera','F','2','Senior',NULL),  ('Ivanova', 'Liza', 'F','1','Middle','2015-11-11'),  ('Kolchina','Natalia','F','1','Junior','2016-01-01'); |
|  |  | Select Lname, Fname from MyCourseTable where Grade='Senior'; |
|  |  | Select \* from MyCourseTable where Gender='M' and Grade='Senior' or Specialty=1; |
|  |  | Select \* from MyCourseTable where YEAR(Start\_date)=2015;  OR  Select \* from MyCourseTable where YEAR(Start\_date)>=2015; if we are talkin for everyone from 2015(not only 2015) |
|  |  | Select \* from MyCourseTable order by Fname DESC, Lname ASC; |
|  |  | *First solution:*  We can add columns like that without changing the order, and then update value;  ALTER TABLE MyCourseTable  -> ADD COLUMN City VARCHAR(255) NOT NULL,,  -> ADD COLUMN Birthday DATE NOT NULL,,  -> ADD COLUMN Salary INT NOT NULL;  Update columns in the same way as below in second solution. |
|  |  | *Second solution:*   1. Or create a new table with needed order:   CREATE TABLE MyCourseTableNew (  ID INT NOT NULL AUTO\_INCREMENT,  Lname VARCHAR(255) NOT NULL,  Fname VARCHAR(255) NOT NULL,  Gender ENUM('M', 'F') NOT NULL,  Specialty VARCHAR(255) NOT NULL,  Grade VARCHAR(255) NOT NULL,  City VARCHAR(255) NOT NULL,  Start\_date DATE NOT NULL,  Birthday DATE NOT NULL,  Salary INT NOT NULL,  PRIMARY KEY (ID));   1. Then insert all old data: INSERT INTO MyCourseTableNew (ID, Lname, Fname, Gender, Specialty, Grade, City, Start\_date, Birthday, Salary)   SELECT ID, Lname, Fname, Gender, Specialty, Grade, NULL, Start\_date, NULL, NULL  FROM MyCourseTable;   1. And then update new values  UPDATE MyCourseTableNew   SET City=  CASE  WHEN ID = 1 THEN 'Voronezh'  WHEN ID = 2 THEN 'Kiev'  WHEN ID = 3 THEN 'Voronezh'  WHEN ID = 4 THEN 'St. Petersburg'  WHEN ID = 5 THEN 'St. Petersburg'  WHEN ID = 6 THEN 'St. Petersburg'  WHEN ID = 7 THEN 'Kherson'  WHEN ID = 8 THEN 'Kherson'  WHEN ID = 9 THEN 'Kiev'  WHEN ID = 10 THEN 'Odessa'  WHEN ID = 11 THEN 'Odessa'  END,  Birthday =  CASE  WHEN ID = 1 THEN '1978-01-03'  WHEN ID = 2 THEN '1987-05-15'  WHEN ID = 3 THEN '1974-02-08'  WHEN ID = 4 THEN '1988-12-13'  WHEN ID = 5 THEN '1980-11-15'  WHEN ID = 6 THEN '1969-05-28'  WHEN ID = 7 THEN '1991-10-21'  WHEN ID = 8 THEN NULL  WHEN ID = 9 THEN '1990-02-11'  WHEN ID = 10 THEN '1980-10-19'  WHEN ID = 11 THEN '1993-05-11'  END,  Salary =  CASE  WHEN ID = 1 THEN 1000  WHEN ID = 2 THEN 1100  WHEN ID = 3 THEN 1000  WHEN ID = 4 THEN 700  WHEN ID = 5 THEN 1000  WHEN ID = 6 THEN 500  WHEN ID = 7 THEN 1200  WHEN ID = 8 THEN 1000  WHEN ID = 9 THEN 1500  WHEN ID = 10 THEN 900  WHEN ID = 11 THEN 800  END  WHERE ID IN (1, 2, 3,4,5,6,7,8,9,10,11);  Or it can be made one by one UPDATE MyCourseTableNew SET City='Voronezh', Birthday ='1978-01-03', Salary=1000 where ID=1; |
|  |  | SELECT \* FROM MyCourseTableNew  WHERE SUBSTRING(Lname, 3, 1) = 'v'  AND CONCAT(Lname, Fname, Gender, Specialty, Grade, City, Start\_date, Birthday, Salary) LIKE '%v%a%'  AND City = 'St. Petersburg'; |
|  |  | SELECT \* FROM MyCourseTableNew  WHERE SUBSTRING(Lname, 2, 1) = 'r'  AND (Lname NOT LIKE '%d%' AND Fname NOT LIKE '%d%');  OR  SELECT \* FROM MyCourseTableNew  WHERE SUBSTRING(Lname, 2, 1) = 'r'  AND CONCAT(Lname , Fname) NOT LIKE '%d%'; |
|  |  | SELECT Fname, Lname, CONCAT('lives in ', City) AS 'Lives in'  FROM MyCourseTableNew  ORDER BY Lname DESC; |
|  |  | SELECT DISTINCT City FROM MyCourseTableNew; |
|  |  | SELECT Fname, Lname FROM MyCourseTableNew  WHERE Grade = 'Senior' AND City <> 'Kiev'; |
|  |  | SELECT \* FROM MyCourseTableNew  WHERE Start\_date IS NULL; |
|  |  | ALTER TABLE MyCourseTableNew  CHANGE Grade Grade2 VARCHAR(255); |
|  |  | SELECT CONCAT(SUM(Salary)) AS TotalIncome, '$' AS Unit  FROM MyCourseTableNew; |
|  |  | UPDATE MyCourseTableNew SET Birthday = '1990-01-01'  WHERE Birthday IS NULL; |
|  |  | SELECT Fname, Lname, DATEDIFF(CURDATE(), Birthday)/365 AS Age  FROM MyCourseTableNew  ORDER BY Birthday ASC  LIMIT 1; |
|  |  | ALTER TABLE MyCourseTableNew  DROP COLUMN Grade2; |
|  |  | SELECT COUNT(\*) AS EmployeeCount FROM MyCourseTableNew  WHERE City = 'St. Petersburg'; |
|  |  | SELECT Fname, Lname FROM MyCourseTableNew  WHERE Salary IN (1000, 500); |
|  |  | SELECT Fname, Lname FROM MyCourseTableNew  WHERE Salary BETWEEN 701 AND 1199; |
|  |  | SELECT Fname, Lname FROM MyCourseTableNew  WHERE Start\_date = (SELECT MAX(Start\_date) FROM MyCourseTableNew); |
| 5 | 7 | SELECT i.intern\_id, i.f\_name, i.l\_name, ispec.specialty\_id  FROM interns i  JOIN interns\_specialty ispec ON i.intern\_id = ispec.intern\_id; |
|  |  | SELECT i.intern\_id, i.f\_name, i.l\_name  FROM interns i  JOIN interns\_specialty ispec ON i.intern\_id = ispec.intern\_id  JOIN specialty s ON ispec.specialty\_id = s.specialty\_id  WHERE s.s\_name = 'QA'  ORDER BY i.l\_name; |
|  |  | SELECT i.intern\_id, i.f\_name, i.l\_name  FROM interns i  JOIN interns\_specialty ispec ON i.intern\_id = ispec.intern\_id  JOIN specialty s ON ispec.specialty\_id = s.specialty\_id  WHERE s.s\_name = 'QA'  ORDER BY i.l\_name; |
|  |  | SELECT i.intern\_id, i.f\_name, i.l\_name  FROM interns i  WHERE i.pr\_st\_date <= DATE\_SUB(CURDATE(), INTERVAL 8 MONTH); |
|  |  | SELECT i.intern\_id, i.f\_name, i.l\_name, DATEDIFF(i.pr\_st\_date, CURDATE()) AS days\_left  FROM interns i  WHERE i.pr\_st\_date > CURDATE(); |
|  |  | SELECT m.mentor\_id, m.f\_name, m.l\_name, YEAR(CURDATE()) - YEAR(m.pr\_st\_date) AS years\_worked  FROM mentors m; |
|  |  | SELECT AVG(m.salary) AS average\_salary  FROM mentors m; |
|  |  | SELECT m.mentor\_id, m.f\_name, m.l\_name, m.salary, m.salary \* 1.07 AS total\_salary  FROM mentors m; |
|  |  | • A UNION B = {1, 2, 3, 4, 5, 6, 7, 8} • A UNION ALL B= {1, 2, 3, 4, 4, 5, 6, 7, 8, 1} • A INTERSECT B ={1, 4} • A EXCEPT B={2, 3} |
|  |  | SELECT f\_name, l\_name  FROM interns  UNION  SELECT f\_name, l\_name  FROM mentors  ORDER BY l\_name; |
|  |  | SELECT m.f\_name AS mentor\_first\_name, m.l\_name AS mentor\_last\_name, MAX(i.pr\_st\_date) AS last\_intern\_practice\_start\_date  FROM mentors m  JOIN interns\_specialty ispec ON m.mentor\_id = ispec.mentor\_id  JOIN interns i ON ispec.intern\_id = i.intern\_id  WHERE i.pr\_st\_date IS NOT NULL  GROUP BY m.f\_name, m.l\_name; |
|  |  | SELECT i.f\_name, i.l\_name  FROM interns i  LEFT JOIN interns\_specialty ispec ON i.intern\_id = ispec.intern\_id  WHERE ispec.intern\_id IS NULL; |
|  |  | SELECT i.f\_name, i.l\_name  FROM interns i  WHERE YEAR(i.pr\_st\_date) = YEAR(CURDATE()) AND i.l\_name LIKE '%a%'; |
|  |  | SELECT s.specialty\_id, s.s\_name  FROM specialty s  JOIN interns\_specialty ispec ON s.specialty\_id = ispec.specialty\_id  GROUP BY s.specialty\_id, s.s\_name  HAVING COUNT(ispec.intern\_id) >= 3; |
| 6 | 8 | CREATE TABLE IF NOT EXISTS Practice\_result (  Intern\_id smallint(5) unsigned NOT NULL,  Mark tinyint(1) NOT NULL,  PRIMARY KEY (Intern\_id)  );  INSERT INTO Practice\_result VALUES  (1, 5),  (2, 4),  (3, 5),  (5, 3),  (6, 5),  (7, 4),  (8, 5),  (9, 4),  (10, 4); |
|  |  | SELECT interns.f\_name, interns.l\_name  FROM interns  JOIN interns\_specialty ON interns.intern\_id = interns\_specialty.intern\_id  JOIN Practice\_result ON interns.intern\_id = Practice\_result.Intern\_id  WHERE interns\_specialty.specialty\_id = '1' AND Practice\_result.Mark = 5; |
|  |  | SELECT mentors.f\_name, mentors.l\_name  FROM mentors  INNER JOIN interns\_specialty ON mentors.mentor\_id = interns\_specialty.mentor\_id  INNER JOIN Practice\_result ON interns\_specialty.intern\_id = Practice\_result.Intern\_id  WHERE NOT EXISTS (  SELECT \* FROM interns  WHERE interns.intern\_id = interns\_specialty.intern\_id  AND NOT EXISTS (  SELECT \* FROM Practice\_result  WHERE Practice\_result.Intern\_id = interns.intern\_id  AND Practice\_result.Mark = 5)); |
|  |  | SELECT specialty.s\_name FROM specialty  JOIN interns\_specialty ON specialty.specialty\_id = interns\_specialty.specialty\_id  JOIN Practice\_result ON interns\_specialty.intern\_id = Practice\_result.Intern\_id  WHERE Practice\_result.Mark = 5  GROUP BY specialty.s\_name  ORDER BY COUNT(\*) DESC  LIMIT 1; |
|  |  | SELECT AVG(Mark) AS average\_mark FROM Practice\_result; |