Please answer the following questions:

1. What is the difference between UNION and UNION ALL?

<https://stackoverflow.com/questions/49925/what-is-the-difference-between-union-and-union-all>

UNION removes duplicates, just like DISTINCT does, UNION ALL does not perform this additional filtering.

1. What substr('12345', -3, 2) will return?

‘34’ – two characters starting with the third character from string’s end (hence negative index).

1. How keyword WITH can be used in a query?

<https://www.geeksforgeeks.org/sql-with-clause/>

First, it appears to be a special Oracle feature (not supported by all database systems) which allows to make temporary “views” – saved and referenced queries that multiple other queries in same Procedure (long piece of code, to be exact) can use.

1. What is the difference between char and varchar2?

<https://stackoverflow.com/questions/20417845/what-is-the-major-difference-between-varchar2-and-char>

Simple char always uses entire memory allocated to it (like char(10) always has 10 symbols, blank included) but varchar only uses up to a maximum allocated, or less, if the value is shorter than X – dedicated max length, my personal in this category is NVarChar – a Multilanguage string.

1. In which cases query “SELECT a FROM t1 UNION SELECT b FROM t2” can give errors?

First thing that comes to mind is that if union members are not separated with parentheses (like (…) union (…) ) programmer can simply introduce logical errors if not careful.

<https://www.mssqltips.com/sqlservertip/5483/issue-with-union-and-columns-with-mismatched-data-types-in-sql-server/>

In this post there are two warnings – first, more related to this question mentions that it is dangerous to try to union table fields of mismatching types, like characters and numbers – obvious problem and crash of query; the second concern is that sometimes unions can get memory costly.

1. What is the difference between the WHERE and HAVING clauses?

Where is used in From clause as a filter, when you have some simple things going on and is usually faster and more efficient than Having because it potentially reduces the amount of information being held by query; Having on the other hand is used in Group By clause when there is some kind of complex calculation going on or a special function like Count(x) and the like when you are forced to “group” data and then you can filter some of it down with Having. Where is also more potent because it is executed before Having in order of things.

TEST for ORACLE SQL

Please write a query from table of people (“PEOPLE”) for following tasks:

1. Find all women whose birthday is in February;
2. Find all people younger than 21 year with double first- or surname (has “-“ in it);
3. Find fist ten people with the most short surnames;
4. Find all people with unique first names;
5. Find all pairs of men whose difference of age is exactly 10 years;
6. Find how many women have first name Helena, Helene, Helenè etc.;
7. Find all salaries less than 1000 and how many people have this salary;
8. Find out if the field CODE has unique values for each person or not;
9. Select every tenth (number 1, 11, 21 etc.) record from table ordered by ID;
10. How to order results in the following way: first salaries 1800 and 2000, then salaries from 1801 to 1999, then all other

I’ll start with setting up a table and it’s values in MySQL local database (sakila is default local database name):

INSERT INTO Sakila.People VALUES (1, '1970-01-01', 'Jill', 'Valentine', 'Female', 500, 'a');

INSERT INTO Sakila.People VALUES (2, '1975-02-01', 'Anne', 'Breeze', 'Female', 600, 'b');

INSERT INTO Sakila.People VALUES (3, '1980-03-01', 'Joana', 'Zanger', 'Female', 700, 'c');

INSERT INTO Sakila.People VALUES (4, '1985-04-01', 'Veronica', 'Jablonski', 'Female', 800, 'd');

INSERT INTO Sakila.People VALUES (5, '1990-05-01', 'Angela', 'Feinberg', 'Female', 900, 'e');

INSERT INTO Sakila.People VALUES (6, '1995-06-01', 'Paulina', 'Burns', 'Female', 1000, 'a');

INSERT INTO Sakila.People VALUES (7, '1995-07-01', 'Suzan', 'Talbott', 'Female', 1100, 'b');

INSERT INTO Sakila.People VALUES (8, '1990-08-01', 'Lady', 'Gowen', 'Female', 1200, 'c');

INSERT INTO Sakila.People VALUES (9, '1985-09-01', 'Tina', 'Sadler', 'Female', 1300, 'd');

INSERT INTO Sakila.People VALUES (10, '1980-10-01', 'Helen', 'Holmgren', 'Female', 1400, 'e');

INSERT INTO Sakila.People VALUES (11, '2010-11-01', 'Mylene', 'Grenier-John', 'Female', 1500, NULL);

INSERT INTO Sakila.People VALUES (12, '1970-12-01', 'Roxane', 'Soyinka', 'Female', 1600, NULL);

INSERT INTO Sakila.People VALUES (13, '1970-01-01', 'Jill', 'Labs', 'Male', 500, 'a');

INSERT INTO Sakila.People VALUES (14, '1975-02-01', 'Gray', 'Hergenroether', 'Male', 600, 'b');

INSERT INTO Sakila.People VALUES (15, '1980-03-01', 'Gwynne', 'Van verth', 'Male', 700, 'c');

INSERT INTO Sakila.People VALUES (16, '1985-04-01', 'Eberhard', 'Soumerai', 'Male', 800, 'd');

INSERT INTO Sakila.People VALUES (17, '1990-05-01', 'Hugh', 'Gastaldi', 'Male', 900, 'e');

INSERT INTO Sakila.People VALUES (18, '1995-06-01', 'Nickolai', 'Behrens', 'Male', 1000, 'a');

INSERT INTO Sakila.People VALUES (19, '2005-06-01', 'Randall', 'Lelacheur', 'Male', 1100, 'b');

INSERT INTO Sakila.People VALUES (20, '1990-08-01', 'Wyn', 'Vandenberg', 'Male', 1200, 'c');

INSERT INTO Sakila.People VALUES (21, '1985-09-01', 'Allen', 'Lipka', 'Male', 1300, 'd');

INSERT INTO Sakila.People VALUES (22, '1980-10-01', 'Calley', 'Vansaun', 'Male', 1400, 'e');

INSERT INTO Sakila.People VALUES (23, '2010-11-01', 'Neron-Wetterstrom', 'Escott', 'Male', 1500, NULL);

INSERT INTO Sakila.People VALUES (24, '1970-12-01', 'Perl', 'Canzanelli', 'Male', 1600, NULL);

The table is named People and has following fields: PeopleId – Int, PK, unique, Birthday - Date, FirstName - NVarChar, SecondName (could be LastName) - NVarChar, Gender - NVarChar, Salary – Int (could be Float) and Code - NVarChar. Some I’ve declared Not Nulls, some not – a subject for in depth debate. I’ve used some online name generators because my imagination in coming up with so many names is poor.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 1990-01-01 | Jill | Valentine | Female | 500 | a |
| 2 | 1975-02-01 | Anne | Breeze | Female | 600 | b |
| 3 | 1980-03-01 | Joana | Zanger | Female | 700 | c |
| 4 | 1985-04-01 | Veronica | Jablonski | Female | 800 | d |
| 5 | 1990-05-01 | Angela | Feinberg | Female | 900 | e |
| 6 | 1995-06-01 | Paulina | Burns | Female | 1000 | a |
| 7 | 1995-07-01 | Suzan | Talbott | Female | 1100 | b |
| 8 | 1990-08-01 | Lady | Gowen | Female | 1200 | c |
| 9 | 1985-09-01 | Tina | Sadler | Female | 1300 | d |
| 10 | 1980-10-01 | Helen | Holmgren | Female | 1400 | e |
| 11 | 2010-11-01 | Mylene | Grenier-John | Female | 1500 |  |
| 12 | 1970-12-01 | Roxane | Soyinka | Female | 1600 |  |
| 13 | 1970-01-01 | Jill | Labs | Male | 500 | a |
| 14 | 1975-02-01 | Gray | Hergenroether | Male | 600 | b |
| 15 | 1980-03-01 | Gwynne | Van verth | Male | 700 | c |
| 16 | 1985-04-01 | Eberhard | Soumerai | Male | 800 | d |
| 17 | 1990-05-01 | Hugh | Gastaldi | Male | 900 | e |
| 18 | 1995-06-01 | Nickolai | Behrens | Male | 1000 | a |
| 19 | 2005-06-01 | Randall | Lelacheur | Male | 1100 | b |
| 20 | 1990-08-01 | Wyn | Vandenberg | Male | 1200 | c |
| 21 | 1985-09-01 | Allen | Lipka | Male | 1300 | d |
| 22 | 1980-10-01 | Calley | Vansaun | Male | 1400 | e |
| 23 | 2010-11-01 | Neron-Wetterstrom | Escott | Male | 1500 |  |
| 24 | 1970-12-01 | Perl | Canzanelli | Male | 1600 |  |

1. Find all women whose birthday is in February:

SELECT FirstName, SecondName, Gender, Birthday

FROM Sakila.People

WHERE Gender = 'Female' AND Month(Birthday) = 02;

|  |  |  |  |
| --- | --- | --- | --- |
| Anne | Breeze | Female | 1975-02-01 |

2. Find all people younger than 21 year with double first- or surname (has “-“ in it):

SELECT FirstName, SecondName, Gender, Birthday

FROM Sakila.People

WHERE FirstName LIKE '%-%' OR SecondName LIKE '%-%'

AND (YEAR(CURRENT\_TIMESTAMP) - YEAR(Birthday) - (RIGHT(CURRENT\_TIMESTAMP, 5) < RIGHT(Birthday, 5))) < 21;

|  |  |  |  |
| --- | --- | --- | --- |
| Mylene | Grenier-John | Female | 2010-11-01 |
| Neron-Wetterstrom | Escott | Male | 2010-11-01 |

3. Find fist ten people with the most short surnames:

SELECT FirstName, SecondName, Gender

FROM Sakila.People

ORDER BY LENGTH(SecondName) ASC LIMIT 10;

(wanted to use TOP here but couldn’t make it work quickly)

|  |  |  |
| --- | --- | --- |
| Jill | Labs | Male |
| Allen | Lipka | Male |
| Paulina | Burns | Female |
| Lady | Gowen | Female |
| Anne | Breeze | Female |
| Joana | Zanger | Female |
| Tina | Sadler | Female |
| Neron-Wetterstrom | Escott | Male |
| Suzan | Talbott | Female |
| Roxane | Soyinka | Female |

4. Find all people with unique first names:

SELECT DISTINCT FirstName

FROM Sakila.People

ORDER BY FirstName;

(notice that there are only 23 results – Jill is a duplicate and therefore skipped once)

|  |
| --- |
| Allen |
| Angela |
| Anne |
| Calley |
| Eberhard |
| Gray |
| Gwynne |
| Helen |
| Hugh |
| Jill |
| Joana |
| Lady |
| Mylene |
| Neron-Wetterstrom |
| Nickolai |
| Paulina |
| Perl |
| Randall |
| Roxane |
| Suzan |
| Tina |
| Veronica |
| Wyn |

5. Find all pairs of men whose difference of age is exactly 10 years:

SELECT main.FirstName, main.SecondName, main.Gender, main.Birthday,

compare.FirstName, compare.SecondName, compare.Gender, compare.Birthday,

DATEDIFF(main.Birthday, compare.Birthday) AS DayDifference

FROM Sakila.People AS main, Sakila.People AS compare

WHERE main.Gender = 'Male' AND compare.Gender = 'Male' AND datediff(main.Birthday, compare.Birthday) = 3653;

(notice though how “exact” might mean “10 years” or “3650 days”, I’ve went for a bit different criteria: must be gap year extra days – those 3, also there is a broad and long topic to be said about naming things in SQL in general and what kind of prefixes to use in given situations – Euromonitor where I’ve worked most of the time had this kind of policy and one Team Lead who left just as I was entering office phase of my employment there had worked a lot to establish those guidelines)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Randall | Lelacheur | Male | 2005-06-01 | Nickolai | Behrens | Male | 1995-06-01 | 3653 |

6. Find how many women have first name Helena, Helene, Helenè etc.:

SELECT FirstName, SecondName, Gender

FROM Sakila.People

WHERE Gender = 'Female' AND FirstName IN ('Helena', 'Helene', 'Helenè', 'Helen');

|  |  |  |
| --- | --- | --- |
| Helen | Holmgren | Female |

7. Find all salaries less than 1000 and how many people have this salary:

SELECT Salary, COUNT(\*)

FROM Sakila.People

WHERE Salary < 1000

GROUP BY Salary;

|  |  |
| --- | --- |
| 500 | 2 |
| 600 | 2 |
| 700 | 2 |
| 800 | 2 |
| 900 | 2 |

8. Find out if the field CODE has unique values for each person or not:

SELECT Code, COUNT(\*) AS Count

FROM Sakila.People

GROUP BY Code

HAVING COUNT(\*) > 1;

(my logic is as follows – let’s see how many distinct values does Code have on all people and if there are any that repeat more than once, null included, if there are any – then not everyone has unique Codes, another note is since I don’t know what Code means, I just use it as a bonus property without context – is it a personal identification code or something else, I have no idea in this context)

|  |  |
| --- | --- |
| a | 4 |
| b | 4 |
| c | 4 |
| d | 4 |
| e | 4 |
|  | 4 |

9. Select every tenth (number 1, 11, 21 etc.) record from table ordered by ID:

SELECT PeopleId

FROM Sakila.People

WHERE PeopleId % 10 = 1;

|  |
| --- |
| 1 |
| 11 |
| 21 |

10. How to order results in the following way: first salaries 1800 and 2000, then salaries from 1801 to 1999, then all other:

SELECT FirstName, SecondName, Salary

FROM Sakila.People

ORDER BY

CASE Salary

WHEN 1400 THEN 0

WHEN 1600 THEN 1

WHEN Salary BETWEEN 1401 AND 1599 THEN 2

ELSE 3

END ASC,

Salary ASC;

(special note is that I am certain that I should use Between And construct and alone it does filter out 1500’s but here it somehow fails to put 1500’s after 1400’s and 1600’s, I’ll leave it at that but I do get what the question is hinting at, also the very last Order By Salary again is for actually ordering all remaining items since otherwise they just stay as their order in table is – 500, 600, … 1500 twice)

|  |  |  |
| --- | --- | --- |
| Helen | Holmgren | 1400 |
| Calley | Vansaun | 1400 |
| Roxane | Soyinka | 1600 |
| Perl | Canzanelli | 1600 |
| Jill | Valentine | 500 |
| Jill | Labs | 500 |
| Anne | Breeze | 600 |
| Gray | Hergenroether | 600 |
| Joana | Zanger | 700 |
| Gwynne | Van verth | 700 |
| Veronica | Jablonski | 800 |
| Eberhard | Soumerai | 800 |
| Angela | Feinberg | 900 |
| Hugh | Gastaldi | 900 |
| Paulina | Burns | 1000 |
| Nickolai | Behrens | 1000 |
| Suzan | Talbott | 1100 |
| Randall | Lelacheur | 1100 |
| Lady | Gowen | 1200 |
| Wyn | Vandenberg | 1200 |
| Tina | Sadler | 1300 |
| Allen | Lipka | 1300 |
| Mylene | Grenier-John | 1500 |
| Neron-Wetterstrom | Escott | 1500 |