**SQL Exercises - answers**

You might want to reload the the tables & add the original data

**1. List the student\_id, firstname and lastname of all the students**

SELECT student\_id, firstname, lastname

FROM Student;

**2. Write an inner join showing student id, surname, module code & marks.**

SELECT s.student\_id, lastname, module\_code, mark

FROM Student s, Grade g

WHERE s.student\_id = g.student\_id;

SELECT s.student\_id, lastname, module\_code, mark

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id);

**3. Find the surname and initials of people who have marks (use DISTINCT).**

The question has asked for initials which is not a field, so is technically impossible. You could just assume firstname was meant, but it is technically possible to give a first initial using SUBSTRING on firstname:

SELECT DISTINCT lastname, SUBSTRING(firstname, 1,1) AS "Initial"

FROM Student s, Grade g

WHERE s.student\_id = g.student\_id AND mark IS NOT NULL;

However, this only gives 2 names...adding student\_id gives all the students who have marks:

SELECT DISTINCT s.student\_id, lastname, SUBSTRING(firstname, 1,1) AS "Initial"

FROM Student s, Grade g

WHERE s.student\_id = g.student\_id AND mark IS NOT NULL;

**4. Find the student name, module code & description of modules they have taken.**

SELECT firstname + ' ' + lastname AS "Student Name", module\_code, m.title

FROM Grade g INNER JOIN Module m ON (g.module\_code = m.code) INNER JOIN Student s ON (s.student\_id = g.student\_id)

ORDER BY s.student\_id;

**5. List all (non null) marks with the name of the student. Order them by marks descending.**

SELECT firstname + ' ' + lastname AS "Student Name", mark

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

WHERE mark IS NOT NULL

ORDER BY mark DESC;

**6. List all nulls mark with the student details.**

Basically same as above...but could be interpretted

SELECT s.student\_id, firstname, lastname, module\_code, mark

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

WHERE mark IS NULL

ORDER BY s.student\_id

**7. List all those students with marks between 60% and 70%**

SELECT s.student\_id, firstname + ' ' + lastname AS "Student Name", mark

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

WHERE mark BETWEEN 60 and 70

ORDER BY s.student\_id;

**8. List all the student names with their average marks**

SELECT s.student\_id, firstname, lastname, AVG(mark)

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

GROUP BY s.student\_id, firstname, lastname;

**9. Show the student id and surname with their number of marks in descending order.**

SELECT s.student\_id, lastname, COUNT(mark)

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

GROUP BY s.student\_id, lastname

ORDER BY COUNT(mark) DESC;

**10. For each student, list their names and marks with the module code and description. (Use aliases to abbreviate table names)**

Basically same as previous...

SELECT firstname + ' ' + lastname AS "Student Name", module\_code AS "Module Code", title AS "Module Description", mark AS "Marks"

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id) INNER JOIN Module m ON (g.module\_code = m.code)

WHERE mark IS NOT NULL

ORDER BY s.student\_id;

**11. List each student whose average marks exceed 60%, list their names & average.**

SELECT s.student\_id, firstname, lastname, AVG(mark)

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

WHERE mark IS NOT NULL

GROUP BY s.student\_id, firstname, lastname

HAVING AVG(mark) > 60;

**12. Add date-of-birth to the Student table, add records & then display all those born before 2000, giving their names & ages**

ALTER TABLE Student

ADD dob DATE;

UPDATE Student

SET dob = '1999-12-31'

WHERE student\_id = 'S103';

UPDATE Student

SET dob = '1998-12-31'

WHERE student\_id = 'S104';

UPDATE Student

SET dob = '2000-01-01'

WHERE student\_id = 'S105';

UPDATE Student

SET dob = '1989-12-31'

WHERE student\_id = 'S107';

Turns out getting the age is not as easy as I thought in SQL Server! This is technically as close as I want to do without trying to work out when their birthday is…

SELECT firstname + ' ' + lastname AS "Name", (YEAR((CAST(GETDATE() AS Date))) - YEAR(dob)) AS “Age”

FROM Student

WHERE YEAR(dob) < 2000;

**13. List the details of students taking either modules in either "PR1" or "IAI", order them by their surname.**

SELECT DISTINCT s.student\_id, firstname, lastname

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

WHERE module\_code = 'PR1' OR module\_code = 'IAI'

ORDER BY lastname;

**14. Find the number of students taking "PR1" or "IAI" (then do the same for those taking "PR1" and "IAI")**

SELECT COUNT(module\_code)

FROM Grade

WHERE module\_code = 'PR1' OR module\_code = 'IAI';

Why can you not use the following?

SELECT COUNT(module\_code)

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

WHERE module\_code = 'PR1' AND module\_code = 'IAI';

SELECT COUNT(module\_code)

FROM Grade

WHERE module\_code = 'PR1' AND student\_id IN (SELECT student\_id

FROM Grade WHERE module\_code = 'IAI');

**15. Display details of all those with marks in module "PR1".**

SELECT s.student\_id, firstname, lastname

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

WHERE module\_code = 'PR1' AND mark IS NOT NULL;

**16. List details of everyone according to ONLY their highest marks, with the largest value at the top.**

SELECT s.student\_id, firstname, lastname, MAX(mark) AS "Max mark"

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

GROUP BY s.student\_id, firstname, lastname

ORDER BY MAX(mark) DESC;

**17. Display the number of students who have received a mark; then the number who have not received a mark.**

This could be interpreted as all the marks given or actual students...

SELECT COUNT(g.student\_id)

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

WHERE mark IS NOT NULL;

OR

SELECT COUNT(student\_id)

FROM Student

WHERE student\_id IN (SELECT g.student\_id

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

AND mark IS NOT NULL);

SELECT COUNT(student\_id)

FROM Student

WHERE student\_id NOT IN (SELECT g.student\_id

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

AND mark IS NOT NULL);

**18. Find the minimum, maximum, and average marks. Give the results aliases.**

SELECT MIN(mark) AS "Minimum", MAX(mark) AS "Maximum", AVG(mark) AS "Average"

FROM Grade;

**19. Find the surname and average marks of all students who get a mark over 70%.**

SELECT s.student\_id, lastname, AVG(mark)

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

WHERE mark IS NOT NULL AND s.student\_id IN (SELECT student\_id FROM Grade

WHERE mark > 70)

GROUP By s.student\_id, lastname;

**20. Calculate the total average mark (call it ‘Average mark’) of all the students.**

SELECT AVG(mark) AS 'Average mark'

FROM Grade;

**21. List the students with an average mark greater than or equal to 40 (Clue: use GROUP BY)**

SELECT s.student\_id, lastname, AVG(mark)

FROM Student s INNER JOIN Grade g ON (s.student\_id = g.student\_id)

GROUP By s.student\_id, lastname

HAVING AVG(mark) >= 40;