

Structured Query Language - SQL - Writing Queries

School IT database with 5 tables. Tables hold personal details, marks for each term, comments and relevant computer terminal details.

In the personal details tables we find grade 10, 11 and 12 students born between 2001 and 2005.

The Primary key is generally the students SD number.

Tables - One for personal details (all grades), one for term marks 2018, one for term marks 2019, one for comments 2019.

One table for the details of individual computer terminals i.e. software installed etc. Primary key text "01", "02" etc

One transactional table (TS) that connects the student's sd number to the terminal where they sit. ** see below.

One student can only sit at one terminal, but an individual terminal has more than one student during the course of the day.

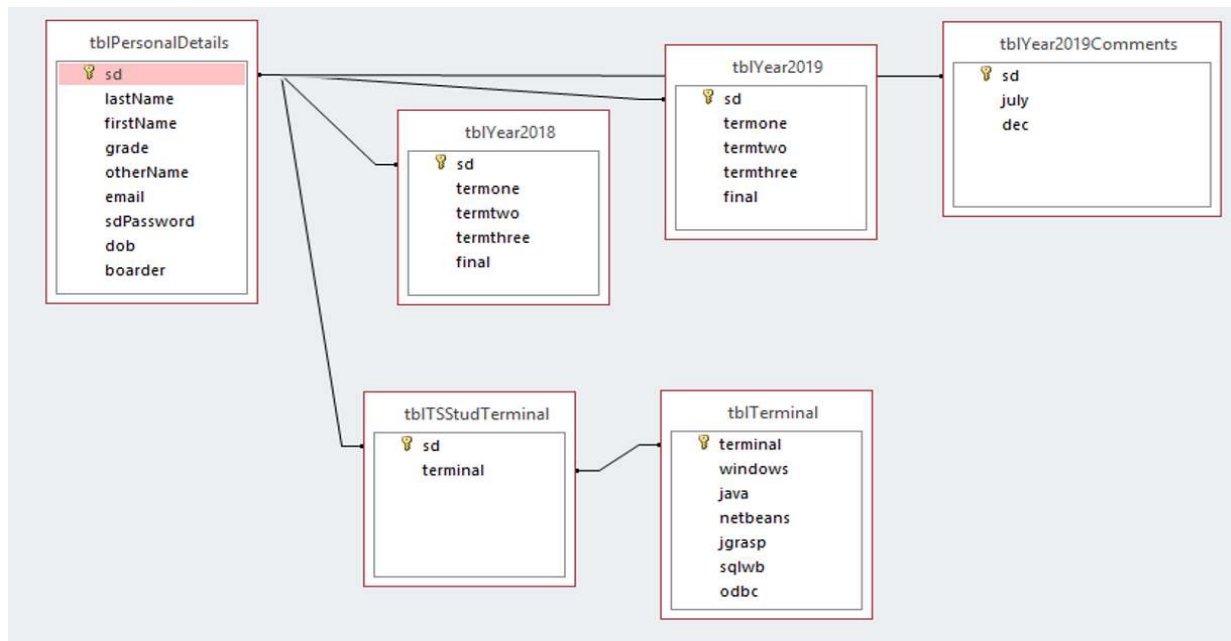
Note that field names are **not** unique - therefore the table name may need to precede the field name in some cases.

e.g. tblPersonalDetails.sd - tblYear2018.sd - tblYear2019.sd etc (referred to as fully qualified)

The relationships between the tables.

Also gives the field names.

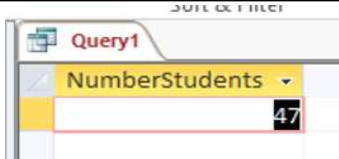
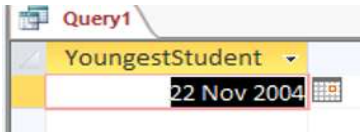
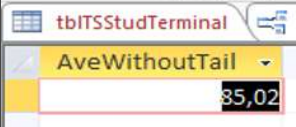
**** The TS Student Terminal table with the SD number matched to the terminal where the student sits.**



sd	terminal
SD10021220	12
SD10021223	11
SD100212231	23
SD10021228	09
SD10021229	08
SD10021238	13
SD10021244	02
SD10021245	04
SD10021247	08
SD10021250	24
SD10021256	08
SD10021259	10
SD10021260	20
SD10021261	19
SD10021271	16
SD10021355	22
SD10021356	09
SD10021358	24
SD10021360	05
SD10021362	08
SD10021363	10
SD10021365	06

One Table Queries	
SELECT firstName, lastName, grade FROM tblPersonalDetails ORDER BY grade, lastName;	Returns the three fields from the personal table, sorted first by grade, and then by lastName
SELECT final from tblYear2019 WHERE sd = "sd10021463";	Returns the final mark only, from the 2019 table that matches the given student SD number (which is stored as text)
SELECT sd from tblYear2019 where sd LIKE "sd100214*";	Returns the student SD numbers that start with "sd100214" but end with any other value (see * wild card character)
SELECT TOP 5 termthree, sd FROM tblYear2019 ORDER BY termthree DESC ;	Returns the top 5 marks from term three 2019. DESC means that the highest marks are at the top. Also returns the SD numbers of the highest achieving students.
SELECT DISTINCT lastName, firstName FROM tblPersonalDetails ORDER BY lastName;	Returns unique last names (with first name) in alphabetical order by last name.
SELECT lastName, firstName FROM tblPersonalDetails WHERE MONTH(dob) = 3 ORDER BY lastName;	Returns students born in March Sorted by last name.
SELECT lastName, firstName, (NOW() - dob) / 365 AS Age FROM tblPersonalDetails ORDER BY lastName;	Returns age, by subtracting the date of birth from today's date. This calculated field is named "Age" and the answer is divided to get years. (Still needs ROUND to get rid of fractions of a year - see below for ROUND)
SELECT lastName, firstName FROM tblPersonalDetails WHERE YEAR(dob) = 2002 OR YEAR(dob) = 2003 ORDER BY lastName;	Returns students born in 2002 or 2003. Sorted by last name.
SELECT lastName, firstName FROM tblPersonalDetails WHERE dob IS NULL ORDER BY lastName;	Returns students whose date of birth is missing from the database

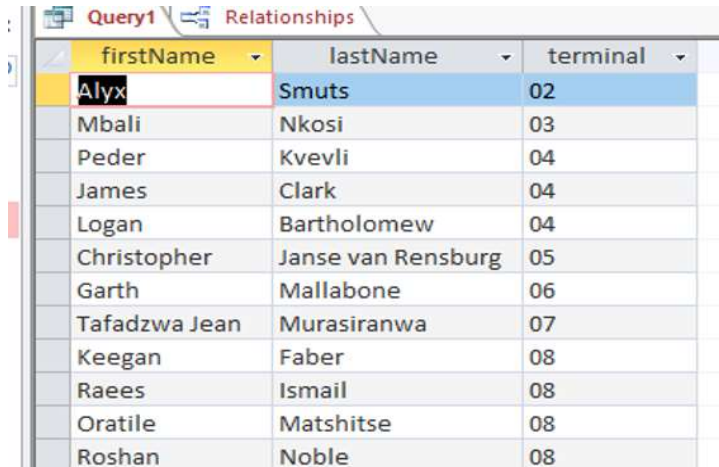
One Table Queries	
SELECT lastName, firstName, grade FROM tblPersonalDetails WHERE LEFT (lastName,1) = "M" ORDER BY grade, lastName;	Returns all students whose last name begins with "M". Sorted first by grade and then by last name.
SELECT AVG ((NOW() - dob) / 365) AS AveAge FROM tblPersonalDetails;	Returns the average age of the students in the database using today's date. The students ages are worked out in days, and then years (/365) Then the average age is determined. This calculated field (derived field) is named "AveAge".
SELECT lastName, firstName FROM tblPersonalDetails WHERE DAY(dob) BETWEEN 1 AND 7 ORDER BY lastname;	Returns all students whose date of birth falls between the 1st day and the 7th day of the month.
SELECT lastName, firstName FROM tblPersonalDetails WHERE DAY(dob) BETWEEN 1 AND 7 AND grade = "11y" ORDER BY lastname;	Returns all students whose date of birth falls between the first day and the 7th day of the month who are in grade 11Y.
SELECT lastName, firstName, grade FROM tblPersonalDetails WHERE email is null AND boarder = yes ORDER BY lastName;	Returns all students who do not have an email address and who board. Board is a "Yes/No" data type.
SELECT lastName, firstName, grade FROM tblPersonalDetails WHERE grade IN ('10X', '10Y', '10Z') ORDER BY grade, lastName;	Returns all students whose grade is mentioned in the list. Quicker than multiple conditions with "AND", especially if the list is long. Sorted by grade and last name.
SELECT TOP 5 lastName FROM tblPersonalDetails ORDER BY LEN (lastName) DESC;	Returns the 5 students with the longest last names.

One Table Queries	
SELECT COUNT(*) AS NumberStudents FROM tblPersonalDetails;	Counts the students using the aggregate function COUNT - (note *) Aggregates only yeild one line, with the requested information. See below on the left .
	
SELECT MAX (dob) AS YoungestStudent FROM tblPersonalDetails;	Using the aggregate function MAX finds the student with the max dob. NOTE: Therefore this is the youngest student. Aggregates only yeild one line, with the requested information. See above on the right .
SELECT SUM (termone) AS TotalTermOne FROM tblYear2019;	Returns the sum of the Term One column from 2019 - only. Names the derived column as "TotalTermOne".
SELECT ROUND (AVG (termone),0) AS AveTermOne FROM tblYear2019;	Returns the average of the Term One column - the average is then rounded to zero decimal places. Names the derived column as "AveTermOne".
SELECT COUNT(*) AS Boarders FROM tblPersonalDetails WHERE boarder = yes;	Counts the number of boarders. Here the aggregate function has a condition under the WHERE clause. Again note the * used with COUNT.
SELECT ROUND(AVG(final),2) AS AveWithoutTail FROM tblYear2019 WHERE final >= 39;	Aggregate AVE gives the average of the final 2019 mark rounded to 2 decimal place. The WHERE clause ignores any mark below 40 when working out the average. See below.
	

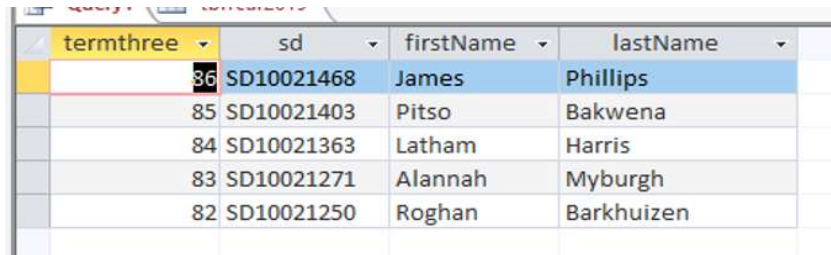
Two Table Queries - You need to explain the JOIN as part of the query code. We use the WHERE clause for this.

```
SELECT firstName, lastName, terminal
FROM tblTSSstudTerminal, tblPersonalDetails
WHERE tblPersonalDetails.sd = tblTSSstudTerminal.sd
ORDER BY terminal;
```

Selects three fields from two different tables. The query is informed about the JOIN - see the WHERE statement. The WHERE statement explains the relationship to the query. See result below on the **left**.



firstName	lastName	terminal
Alyx	Smuts	02
Mbali	Nkosi	03
Peder	Kvevli	04
James	Clark	04
Logan	Bartholomew	04
Christopher	Janse van Rensburg	05
Garth	Mallabone	06
Tafadzwa Jean	Murasiranwa	07
Keegan	Faber	08
Raees	Ismail	08
Oratile	Matshitse	08
Roshan	Noble	08



termthree	sd	firstName	lastName
86	SD10021468	James	Phillips
85	SD10021403	Pitso	Bakwena
84	SD10021363	Latham	Harris
83	SD10021271	Alannah	Myburgh
82	SD10021250	Roghan	Barkhuizen

```
SELECT TOP 5 termthree, tblPersonalDetails.sd, firstName,
lastName
FROM tblYear2019, tblPersonalDetails
WHERE tblYear2019.sd = tblPersonalDetails.sd
ORDER BY termthree DESC;
```

Returns the top 5 students in 2019 term three with their names - this comes from two different tables. The WHERE statement explains the JOIN to the query. ORDER BY DESC ensures that the highest marks are at the top. The field "sd" appears in two different tables, therefore you need to precede the field name with table name. See results above on the **right**.

```
SELECT SUM(termone)
FROM tblPersonalDetails, tblYear2019
WHERE tblPersonalDetails.sd = tblYear2019.sd
AND boarder = yes AND grade IN ('10X', '10Y', '10Z');
```

Adds the marks from term one, but only if you are a boarder **and** you are in grade 10. Marks are in one table and the grade and boarder status is in another table. Here, the WHERE clause explains both the JOIN as well as the condition.

Two/Three Table Queries - You explain the JOIN as part of the query code. We use the WHERE clause for this.

```
SELECT firstName, lastName, termone, termtwo
FROM tblPersonalDetails, tblYear2018
WHERE tblPersonalDetails.sd = tblYear2018.sd
AND firstName = "Latham" AND lastName = "Harris";
```

Returns Latham Harris' term one and term two mark from 2018.
His name is found in the one table but his marks are in a different table. The WHERE clause is used for the JOIN and for the condition - the name in this case.

Three tables

```
SELECT firstName, lastName,
    tblYear2018.termone, tblYear2018.termtwo,
    tblYear2019.termone, tblYear2019.termtwo
FROM tblPersonalDetails, tblYear2018, tblYear2019
WHERE tblPersonalDetails.sd = tblYear2018.sd
AND tblPersonalDetails.sd = tblYear2019.sd
AND firstName = "Latham" AND lastName = "Harris";
```

Returns Latham Harris' 2018 term one and term two marks, as well as his 2019 marks for term one and term two.
His name is found in the 1st table; his 2018 marks are in a 2nd table and 2019 marks in a 3rd table.

- 1) Start off by listing all the fields you need (fully qualified)
- 2) List the tables you need.
- 3) Explain the joins to the query.

(table one = table two = table three) is **not** allowed in SQL or even in Java
(table one = table two) **AND** (table one = table three) is allowed.
See results below.

firstName	lastName	tblYear2018	tblYear2018	tblYear2019	tblYear2019
Latham	Harris	70	68	59	69

```
SELECT firstName, lastName,
    (termone + termtwo + termthree) As YearTotal
FROM tblPersonalDetails, tblYear2019
WHERE tblPersonalDetails.sd = tblYear2019.sd
ORDER BY (termone + termtwo + termthree) DESC;
```

This query totals the three term marks together.
The output has to be ordered by the addition expression as being "derived" it does not have a field name of its own.
See output below.

firstName	lastName	YearTotal
Alannah	Myburgh	218
Latham	Harris	212
Roghan	Barkhuizen	212
Martin	Muller	211
Ronewa Nicke	Fakude	206
Kristen	Banabotlhe	206
Oteng	Dintwile	206