Applied 9 SQL Part I - Basic SQL

Applied 9 SQL Part I - Basic SQL

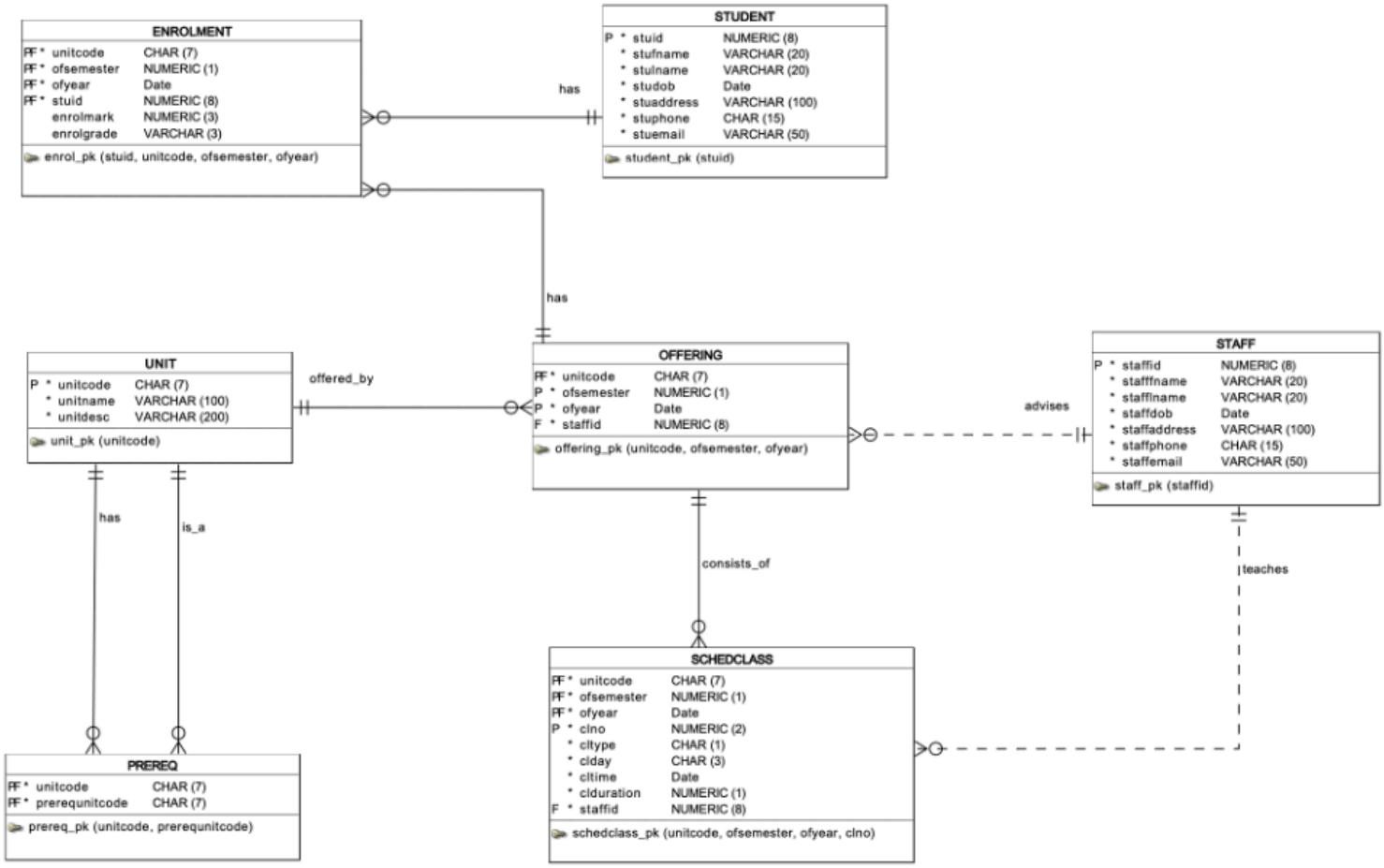
After completing this activity, you should be able to:

- interpret a graphical representation of a relational database
- code simple SQL statements on a single table
- code SQL statements that use rows from more than a single table using different types of ANSI standard JOIN operations
- code SQL SELECT statements to select rows based on different conditions
- use ORACLE's date data type in SQL statements correctly
- define an alias for tables and columns
- sort the retrieved data into different orders via SQL ORDER BY
- use select statements in DML

This activity supports unit learning outcomes 1 and 4.

A9-1 The UNIVERSITY Database, Handling Dates and Strings

The exercises you will complete in this module will make use of the UNIVERSITY set of tables available via our Oracle database. The figure below depicts a data model for the UNIVERSITY set of tables:



In the Monash Oracle database, this UNIVERSITY set of tables has been created under the user "UNI". **In Oracle to access another user's tables you must preface the table name with the username**, thus, to use these tables you need to add the prefix "UNI" to the table names that you use in an SQL statement. For example, if you want to show the table details/structure you need to write:

```
desc uni.unit;
```

or for more detailed information

```
info uni.unit;
```

Note that the info command is not available under ORDS.

If you want to retrieve data from the UNIT table, you need to write:

```
SELECT
    unitcode,
    unitname
```

```
FROM
    uni.unit;
```

and not:

```
SELECT
    unitcode,
    unitname
FROM
    unit;
```

Without a prefix on the table name the select looks for a table of this name in your account.

SQL Statements and Date Format

Note the following:

- The Oracle date data type contains both date and time, however, you can choose to use just a date, just a time, both or parts of a date depending on the format strings used
- **to_date**: converts from a string to a date according to a format string

```
SELECT
    stuid, stufname, stulname
FROM
    uni.student
WHERE
    studob < TO_DATE('30/Apr/1992', 'dd/Mon/yyyy')
ORDER BY
    stuid;
```

- **to_char**: converts from a date to a string according to a format string

```
SELECT
    to_char(sysdate, 'dd/Mon/yyyy hh24:mi:ss') AS server_date
FROM
    dual;

SELECT
    to_char(sysdate+10/24, 'hh24:mi:ss') AS server_time_plus_10_hrs
FROM
    dual;
```

For this unit you are REQUIRED to make use of to_date/to_char whenever working with any date type attribute.

The Oracle documentation links are:

- Format models
- to_date

- to_char

Comparing Strings in SQL Statements

For this unit when you are asked to match a given string in the database, the string will be indicated in italics.

For example, using our DRONE model from the workshops: list all the drone types which have been manufactured by *DJI Da-Jiang Innovations*. The manufacturer's name string you have been provided in this question (*DJI Da-Jiang Innovations*) **must** be used in your SQL code **EXACTLY as provided** (same spacing, case etc). From our previous discussions you should remember that we cannot know the case of this string in the database, so here we will need a where clause such as

```
WHERE UPPER(manuf_name) = UPPER('DJI Da-Jiang Innovations')
```

Of course, you could also choose to use lower on both sides. The key point to note is that you must use *DJI Da-Jiang Innovations* as presented in the question.

If the question provides a value which is made up of two attributes in the database, you may divide the provided string as required. For example: list all the drones which the employee named *Malika Casey* has booked out for rental (for this case only, you may assume there is only one employee with such a name). In the drone model, EMPLOYEE consists of two attributes emp_fname and emp_lname, a suitable where clause would be

```
WHERE UPPER(emp_fname) = UPPER('Malika') and
      UPPER(emp_lname) = UPPER('Casey')
```

Again here, you could also choose to use lower on both sides.

A9-2 Retrieving data from a single table

Download **applied9_sql_basic.sql** from below:

applied9_sql_basic.sql

Place this file in your working directory in the App09 folder. Write your answers for the questions below in the provided area. Make sure that you include a semicolon ';' at the end of each select statement. Test the select statements one by one by clicking the Run Statement icon.

TASKS

A1. List all units and their details. Order the output by unit code.

A2. List the full student details for those students who live in *Caulfield*. Order the output by student id.

A3. List the full student details for those students who have a surname starting with the letter *M*. In the display, rename the columns stufname and stulname to firstname and lastname. Order the output by student id.

A4. List the student's id, surname, first name and address for those students who have a surname starting with the letter *S* and first name which contains the letter *i*. Order the output by student id.

A5. Assuming that a unit code is created based on the following rules:

- The first three letters represent the faculty abbreviation, eg. *FIT* for the Faculty of Information Technology.
- The first digit of the number following the letter represents the year level. For example, FIT2094 is a unit code from the Faculty of IT (*FIT*) and the number 2 refers to a second year unit.

List the unit details for all first-year units in the Faculty of Information Technology. Order the output by unit code.

A6. List the unit code and semester of all units that are offered in 2021. Order the output by unit code, and within a given unit code order by semester. To complete this question, you need to use the Oracle function to_char to convert the data type for the year component of the offering date into text. For example, to_char(ofyear,'yyyy') - here we are only using the year part of the date.

A7. List the year and unit code for all units that were offered in either semester 2 of 2019 or semester 2 of 2020. Order the output by year and then by unit code. To display the offering year correctly in Oracle, you need to use the to_char function. For example, to_char(ofyear,'yyyy').

A8. List the student id, unit code and mark for those students who have failed any unit in semester 2 of 2021 (a fail is where the mark < 50). Order the output by student id then order by unit code.

A9. List the student id for all students who have no mark and no grade in *FIT3176* in semester 1 of 2020. Order the output by student id.

Important

You need to get into the habit of establishing this as a standard workflow:

- before modifying any file/s, pull at the start of your working session, work on the activities you wish to/are able to complete during this session, add all(stage), commit changes and then push the changes back to the FIT GitLab server.

A9-3 Retrieving data from multiple tables

For this unit, students are REQUIRED to use ANSI JOINs, *placing the join in the where clause is not acceptable and will be* ***marked as incorrect for all assessment purposes****.*

You are free to use:

- JOIN … ON …
- JOIN …. USING …, or
- NATURAL JOIN

in your ANSI join clauses. We suggest **JOIN … ON …** since this is the general form which always works (the other two forms depend on declared PK/FK's and/or attribute naming).

TASKS

Continue working in the file **applied9_sql_basic.sql.** Write your answers for the questions below in the provided area. Make sure that you include a semicolon ';' at the end of each select statement. Test the select statements one by one by clicking the Run Statement icon.

B1. List all the unit codes, semesters and name of chief examiners (ie. the staff who is responsible for the unit) for all the units that are offered in 2021. Order the output by semester then by unit code.

B2. List all unit codes, unit names and their year and semester of offering. Order the output by unit code then by offering year and then semester.

B3. List the student id, student name (firstname and surname) as one attribute and the unit name of all enrolments for semester 1 of 2021. Order the output by unit name, within a given unit name, order by student id.

B4. List the id and full name of all students who have marks in the range of 80 to 100 in *FIT9132* semester 2 of 2019. Order the output by full name. If there are more than one student with the same name, order them based on their id.

B5. List the unit code, semester, class type (lecture or tutorial), day, time and duration (in minutes) for all units taught by *Windham Ellard* in 2021. Sort the list according to the unit code, within a given unit code, order by offering semester.

B6. Create a study statement for *Brier Kilgour*. A study statement contains unit code, unit name, semester and year the study was attempted, the mark and grade. If the mark and/or grade is unknown, show the mark and/or grade as 'N/A'. Sort the list by year, then by semester and unit code. You may assume that there is only one student named *Brier Kilgour.*

B7. List the unit code, unit name and the unit code and unit name of the prerequisite units for all units in the database which have prerequisites. Order the output by unit code and prerequisite unit

code.

B8. List the unit code and unit name of the prerequisite units of the *Introduction to data science* unit. Order the output by prerequisite unit code.

B9. Find all students (list their id, firstname and surname) who have received an *HD* for *FIT2094* in semester 2 of 2021. Sort the list by student id.

B10. List the student's full name, unit code for those students who have no mark in any unit in semester 1 of 2021. Sort the list by student full name.

Important

You need to get into the habit of establishing this as a standard workflow:

- before modifying any file/s, pull at the start of your working session, work on the activities you wish to/are able to complete during this session, add all(stage), commit changes and then push the changes back to the FIT GitLab server.