School of Information Technology
Department of Computer Science

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Faculty of Engineering, Built Environment and Information Technology

COS326 Database Systems: Practical 2 PostgreSQL ORDBMS

Handout date: 6 September 2021 Due date: 17 September 2021 Marks: 50

Objectives

1. Get exposure to the PostgreSQL object-relational DBMS (ORDBMS).
2. Learn how to implement domains, types (UDTs), sequences, tables, functions and table inheritance in an ORDBMS.
3. Appreciate the differences between a relational DBMS, object-oriented DBMS and object-relational DBMS.

You are expected to have completed Tutorial 2 on PostgreSQL before you start on this practical exercise. When you are done:

1. You must submit files, named:
   a. **UML.pdf** which contains the class diagram for the database. Your name and student number must appear in this document.
   b. **CreateStatements.sql** which contains all statements necessary to create the database 'objects' i.e. domains, types, sequences, tables and functions.
   c. **InsertQueries.sql** which contains all statements that add to the content of the database (INSERT statements).
   d. **SelectQueries.sql** which contains all statements that provide reports from the database (SELECT statements)
   e. Compress the above documents into an archive (zip file) and upload it to ClickUP **before** the due date/time. The file name for the archive must have your student number as part of the file name, e.g. xxxxx-prac2.zip where xxxxx is your student number.

2. Detailed information on how the marking will be conducted for this practical exercise will appear on ClickUP next week.

Task 1: Domains, sequences, user-defined types and table inheritance [30 marks]

The Student Records Unit of a university needs to keep records of all **students**: **undergraduates** and **postgraduates,** as well as **degree programs,** and **courses.** For each **student** the student number, name (title, first name, surname), date of birth, degree program, and the year of study (1st year, 2nd year, etc), should be recorded. For an **undergraduate student**, the courses currently registered for, should additionally be recorded. For a **postgraduate student**, the category (part time or full time), and supervisor (title, first name, surname), should additionally be recorded. For each **degree program** the degree name, number of years, department, and faculty should be recorded. For each **course**, the course code, course name, course credits, and department should be recorded.

The above information is summarised as follows:

Entities, attributes and functions:

Entity	**Attributes (values)**	**functions (and examples of return values)**
Student	student number (of six numeric characters), full names (title, first name, surname), where title is one of : Ms, Mev, Miss, Mrs, Mr, Mnr), date of birth, degree code (e.g. BSc), year of study (e.g. 1, 2, 3, etc)	*personFullNames* as 'title first name surname' (e.g. Ms Good Student) (e.g. Mr Serious Student) *ageInYears* (e.g. 22 years) (computed from date of birth)
Undergraduate (is-a student)	**additional attributes:** courseRegistration (an array of course codes, e.g. ['COS301', 'COS302', 'COS326',….])	*isRegisteredFor(…)* returns true if the student is registered for the course with the specified course code. *isFinalYearStudent(…)* returns true if the student is in the final year of his/her degree program.
Postgraduate (is-a student)	**additional attributes:** category (part time or full time), supervisor (title, first name, surname) where title is one of : Ms, Mev, Miss, Mrs, Mr, Mnr, Dr, Prof	*isFullTime(..), isPartTime(..)* The *is..* functions return a Boolean value (true or false) *personFullNames* as 'title first name surname' (e.g. Prof. Very Serious)
DegreeProgram	degree code, (e.g. BSc) degree name (e.g. Bachelor of Science), number of years, faculty	
Course	course code (e.g. COS326), course name (e.g. Database Systems), department (e.g. Computer Science), credits	

To be done:

1. Create a UML class diagram to show the above relationships (2 marks)
2. Identify the classes that should be implemented as tables.
3. Write SQL statements to create:
   a. all necessary domains, enum and structured types (CREATE DOMAIN, CREATE TYPE …) (5 marks)
   b. a sequence that will generate a surrogate (primary) key for each table. (CREATE SEQUENCE). You must specify the start value for the sequence. (3 marks)
   c. the tables to store the objects in the class hierarchy (hierachies) (CREATE TABLE … (INHERITS)) (8 marks)
   d. all the functions for the tables. For functions whose input arguments are UDTs and the output is text, consider using the CAST operator. HINT: Use the following syntax for all functions:

```
CREATE FUNCTION functionname(argtype1, argtype2, .. ) RETURNS returntype AS $$
        SELECT expression to compute AS functionname;
$$ LANGUAGE SQL;
```

(12 marks)

4. **Create a database in PostgreSQL called *studentsDB* and run all the SQL statements in (3) above to create the database 'objects'.** As you create the database 'objects' in pgAdminIII, right click on your database in the object browser and select 'Refresh' so that you can see the created 'objects'. **Note:** marks for part (3) will only be awarded if the database 'objects' actually get created.

Task 2: Inserting data into the Database tables [6 marks]

Use the *INSERT INTO* SQL statement to add the following data into the database. Execute some SELECT statements to confirm that you entered the data correctly.

	Attribute values: note that the values of attributes **… key** are generated by the SEQUENCEs that you created								
Degree Program	**degree key**	**degree code**	**degree name**	**number of years**	**faculty**				
		BSc	Bachelor of Science	3	EBIT				
		BIT	Bachelor of IT	4	EBIT				
		PhD	Philosophiae Doctor	5	EBIT				
Course	**course key**	**course code**	**course name**	**department**	**credits**				
		COS301	Software Engineering	Computer Science	40				
		COS326	Database Systems	Computer Science	20				
		MTH301	Discrete Mathematics	Mathematics	15				
		PHL301	Logical Reasoning	Philosophy	15				
Under graduate	student key	student number	student name (title, fname, surname)	date of birth (dd-mm-yyyy)	degree code	year of study	courseRegistration		
		140010	choose title & names	10-01-1996	BSc	3	COS301, COS326, MTH301		
		140015	choose title & names	25-05-1995	BSc	3	COS301, PHL301, MTH301		
		131120	choose title & names	30-01-1995	BIT	3	COS301, COS326, PHL301		
		131140	choose title & names	20-02-1996	BIT	4	COS301, COS326, MTH301, PHL301		
Postgraduate	student key	student number	student name (title, fname, sname)	date of birth	degree code	year of study	category	supervisor (title,fname, sname)	
		101122	choose title & names	15-06-1987	PhD	2	full time	choose title & names	
		121101	choose title & names	27-04-1985	PhD	3	part time	choose title & names	

Task 3: Querying the Database [14 marks]

Write SELECT statements to provide the reports described in the following table:

Required report: List of all	Columns in report, e.g.	Marks
1. students personal details (you must use the *personFullNames* and ageInYears functions)	Data Output | Explain | Messages | History studentkey integer | studentnumber character(6) | personfullnames text | ageinyears double precision 1 | 301 | 140010 | Miss Good Student | 20 2 | 302 | 140015 | Mr Serious Guy | 21	2
2.undergraduate students registration details (you must use the *personFullNames* function)	Data Output | Explain | Messages | History studentkey integer | studentnumber character(6) | personfullnames text | degreecode text | yearofstudy smallint | courseregistration text[] 1 | 301 | 140010 | Miss Good Stude: | BSc | 3 | {COS301,COS326,MTH301} 2 | 302 | 140015 | Mr Serious Guy | BSc | 3 | {COS301,PHL301,MTH301}	2
3. postgraduate students registration details (you must use the *personFullNames* function)	Data Output | Explain | Messages | History studentkey integer | studentnumber character(6) | personfullnames text | degreecode text | yearofstudy smallint | category categorytype | personfullnames text 1 | 305 | 101122 | Miss Future Astronaut | PhD | 2 | Full time | Prof Famous Scientist 2 | 306 | 121101 | Mr Determined Research | PhD | 3 | Part time | Dr Good Adviser	2
4.undergraduate students registration details for final year students (you must use the *isFinalYear* function)	Data Output | Explain | Messages | History studentkey integer | studentnumber character(6) | personfullnames text | degreecode text | yearofstudy smallint | courseregistration text[] 1 | 301 | 140010 | Miss Good Student | BSc | 3 | {COS301,COS326,MTH301} 2 | 302 | 140015 | Mr Serious Guy | BSc | 3 | {COS301,PHL301,MTH301}	3
5.undergraduate students registration details for students registered for , e.g. COS326 (you must use the *isRegisteredFor* functions)	Data Output | Explain | Messages | History studentkey integer | studentnumber character(6) | personfullnames text | degreecode text | yearofstudy smallint | courseregistration text[] 1 | 301 | 140010 | Miss Good Student | BSc | 3 | {COS301,COS326,MTH301} 2 | 303 | 131120 | Miss Very Smart | BIT | 3 | {COS301,COS326,PHL301}	3
6. full-time postgraduate students registration details (you must use the *isFullTime*function	Data Output | Explain | Messages | History studentkey integer | studentnumber character(6) | personfullnames text | degreecode text | yearofstudy smallint | category categorytype | personfullnames text 1 | 305 | 101122 | Miss Future Astronaut | PhD | 2 | Full time | Prof Famous Scientist	1
7. part-time postgraduate students registration details (you must use the *isPartTime*function	Data Output | Explain | Messages | History studentkey integer | studentnumber character(6) | personfullnames text | degreecode text | yearofstudy smallint | category categorytype | personfullnames text 1 | 306 | 121101 | Mr Determined Researcher | PhD | 3 | Part time | Dr Good Adviser	1

Additional instructions

In order to create the *.sql files, proceed as follows:

a. Open the pgAdminIII query tool.

b. For the file **CreateStatements.sql :** copy all the CREATE statements into the query pane and select Save. Take careful note of the quotation marks in your queries. They must be the 'pure text quotation marks' and not the ‘MS Word quotation marks’. Accept the .sql extension and type in the file name.

b. For the file **InsertQueries.sql** : Open a new window and copy all the INSERT statements into the query pane and select Save. Again, accept the .sql extension and type in the file name.

c. For the file **SelectQueries.sql :** Open a new window and copy all the SELECT statements into the query pane and select Save. Again, accept the .sql extension and type in the file name.

d. Test that your queries are working by doing the following: (1) delete the database you created (2) create the database again (3) open the query tool. (4) open the file Create Statements.sql and click on the execute query button. Repeat this for the InsertQueries.sql and then the SelectStatements.sql file.