INFS2200/INFS7903 Assignment 1, 2025

Total Marks	30 marks (20% of the course)
Due Date	**4:00 pm, 12th September 2025**
What to submit	Report file and SQL script file
Where to submit	Electronic submission via Blackboard Ultra Course Website
Assignment Version	1.1 (released 23rd August)

The goal of the project assignments is to gain practical experience in applying several database management concepts and techniques using the PostgreSQL DBMS. This assignment mainly focuses on ensuring database semantics using various integrity constraints. Your main task is to first populate your database with appropriate data, then design, implement, and test the appropriate queries to perform the tasks explained in the next sections.

You must work on this project **individually.** Academic integrity policies apply. Please refer to *3.60.04 Student Integrity and Misconduct* of the University Policy for more information.

This document is in three sections: Section 1 describes the database schema for the assignment and provides instructions on downloading the script file needed to create and populate the database. Section 2 describes the tasks to be completed for this assignment. The final section describes the submission guidelines and marking scheme.

Enjoy the project!

Section 1: The **SalesDB** Database

You are a developer working on SalesDB.

The SALES database (Figure 1) captures the sales information in a company that provides various IT services. The database includes four tables: CUSTOMER, SALES, STAFF, and DEPT.

- CUSTOMER stores information about all the company's clients.
- SALES keeps track of the service purchases made by the clients.
- STAFF stores information about the employees who work directly with the clients and serve their purchase requests.
- Staff work in different departments, and the information about these departments is stored in the DEPT table.

Database Constraints

Number	Constraint Name	Table.Column	Description
1	PK_STAFFNO	STAFF.StaffNo	StaffNo is the primary key of STAFF
2	PK_DEPTNO	DEPT.DeptNo	DeptNo is the primary key of DEPT
3	PK_SALENO	SALES.SaleNo	SaleNo is the primary key of SALES
4	PK_CUSTOMERNO	CUSTOMER.CustomerNo	CustomerNo is the primary key of CUSTOMER
5	UN_DNAME	DEPT.DName	DName values are unique
6	CK_STAFFNAME	STAFF.StaffName	StaffName must not be empty (not null)
7	CK_DNAME	DEPT.DName	DName must not be empty (not null)
8	CK_CNAME	CUSTOMER.CName	CName must not be empty (not null)
9	CK_RECEIPTNO	SALES.ReceiptNo	ReceiptNo must not be empty (not null)
10	CK_AMOUNT	SALES.Amount	Amount must be a positive value
11	CK_POSITION	STAFF.Position	Position must be one of the following: 'Group Manager', 'Group Assistant', 'Group Member', 'Team Leader', or 'Branch Manager'
12	CK_SERVICETYPE	SALES.ServiceType	Service type must be one of the following: 'Software Installation', 'Software Repair', 'Training', 'Consultation' or 'Data Recovery'
13	CK_PAYMENTTYPE	SALES.PaymentType	Payment type must be one of the following: 'Debit', 'Cash', or 'Credit'
14	CK_GST	SALES.GST	GST must be either 'Yes' or 'No'
15	FK_DEPTNO	STAFF.DeptNo	STAFF.DeptNo refers to DEPT
16	FK_STAFFNO	SALES.ServedBy	SALES.ServedBy refers to STAFF
17	FK_CLIENTNO	SALES.CustomerNo	SALES.CustomerNo refers to CUSTOMER

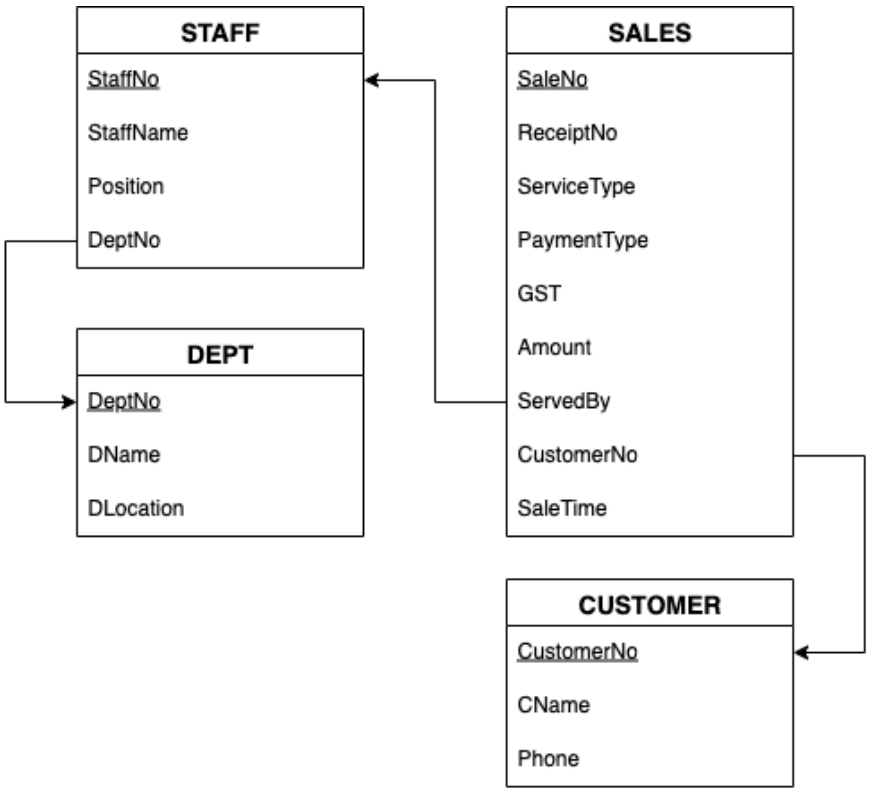

Figure 1: *Database schema diagram.*

Setup

Before starting work on this assignment, please ensure that you the run the setup file **`a1-2025-salesdb.sql`** with the **`\i`** command (it may take a minute).

The setup file is available from Blackboard and on Manta storage:
`$ mget -O /infs2200/stor/data/assignments/1/a1-2025-salesdb.sql`

Section 2: Assignment 1 Tasks

Question 1 – Constraints

Q1.1. After running the script file, you will notice that only some of the constraints listed in Table 1 were created. Write a SQL statement to find out which constraints have been created on the four tables STAFF, DEPT, SALES, and CUSTOMER.

Q1.2. Write the SQL statements to create all the missing constraints.

Question 2 – Triggers

Q2.1. Assume that SaleNo should be automatically populated when a new purchase is made by clients. Write an SQL statement to create a sequence object to generate values for this column. The sequence, named PNO_SEQ, should start from 10,000 and increment by 1.

Q2.2. Write a SQL statement to create an PostgreSQL user-defined function called UDF_BI_PNO and trigger called BI_PNO that binds the sequence object PNO_SEQ to the SaleNo column, i.e., the function/trigger populates values of PNO_SEQ to the SaleNo column when a new purchase is made.

Q2.3. The company's top customer is the one who has purchased the most, i.e., the one with the highest total purchase amount among all the company's customer. Write SQL to create a PostgreSQL user-defined function called UDF_TOP_DISCOUNT and trigger called TOP_DISCOUNT that applies a 15% discount (i.e., 15% reduction to the purchase amount) to any new purchases made by the top customer.

(**Note**: Your function/trigger should not hardcode the top customer, since the top customer could change when more purchases are made by other customers.)

Q2.4. The 'SALES - Sunshine' department has unfortunately run into a technical issue and is temporarily unable to process any 'Credit' or 'Debit' transactions. As a result, it only accepts 'Cash' transactions. Besides, the department is now offering a 30% discount on 'Data Recovery' service. Write SQL to create a PostgreSQL user-defined function UDF_SUNSHINE_DEPT and trigger SUNSHINE_DEPT that will:

(1) set the PaymentType to 'Cash' for any new purchases where the customer is served by an employee of this department;

(2) if the ServiceType is 'Data Recovery', give the customer a 30% discount. This discount is exclusive to the 'SALES - Sunshine' department.

(**Note**: Your function/trigger should not hardcode the DeptNo or StaffNo.)

Q2.5. The business operates in several cities. As a sanity check, management would like to prevent a customer from making a purchase when they have already made one at a *different location* in the 5 minutes prior to the new purchase. Additionally, the business premises are only open to the public between 6am and 9pm. No purchases are to be allowed before 6am or after 9pm, unless the sale is made by a Manager.

Write SQL to create a PostgreSQL user-defined function called UDF_TIME_CHECK and trigger called TIME_CHECK implementing all these business requirements by preventing certain insertions on the SALES table.

(**Note**: don't use *NOW()* or similar – all the information is in the table or the new insert.)

Section 3 - Deliverables and Marking

There are 30 marks for functionality as listed in the task description.

Your assignment code will be tested with insertions and selections when relevant.

When your code must prevent an insertion, you should expect testing for false positives and false negatives.

Triggers will be disabled when not being tested. Triggers and user-defined functions must be created without compilation errors.

Marking Scheme

Question	Marks	Specific Criteria
Q1.1	2	Single SQL query listing all constraints on the four tables.
Q1.2	7	Write only one SQL statement per constraint created. Correctness will be tested using several INSERT & SELECT statements.
Q2.1	2	Sequence created with the correct name and semantics.
Q2.2	3	Correctness will be tested using several INSERT & SELECT statements.
Q2.3	6	Correctness will be tested using several INSERT & SELECT statements. No hard-coding of the top customer.
Q2.4	5	Correctness will be tested using several INSERT & SELECT statements. No hard-coding of DeptNo or StaffNo.
Q2.5	5	Correctness will be tested using several INSERT & SELECT statements.

Submission

Submit **two** documents to Blackboard by the due date (**4pm, 12th September 2025**).

Late submissions will be penalised unless you are approved for an extension (refer to Section 5.3 of the ECP).

Submit a **script** file titled **`4xxxxxxxx_A1.sql`** (plain text with **`.sql`** extension) containing all SQL code you wrote for this assignment. Please do **not** include any `CREATE DATABASE`, `CREATE TABLE` or `psql` commands, as these will interfere with the testing procedure.

Submit a **PDF** file titled **`4xxxxxxxx_A1.pdf`** containing a screenshot of your query/queries for Q1.1 and all results. Also, as a backup to the script file, include a screenshot of each constraint, trigger, function and sequence being created successfully.

Replace **4xxxxxxxx** with your student number for both files.

Your script file must be executable with the **`\i`** command in **`psql`**.

Hint: *in Blackboard Ultra, remember to actually submit and not just save as a draft! Make sure to get a* ***submission receipt****!*

END OF ASSIGNMENT 1

---ooo000O000ooo ---