

Figure 1 CORNER Shop – download from Brightspace

1. Upload files.
   1. Connect to the University PostgreSQL server (147.252.250.51).
   2. Upload and run the file CreateCornerShop2324.sql. This will create and populate the tables shown above.
2. Write a procedure to allow a user to add a sale, with the following steps:
   1. Pass the stock\_code (p\_stock\_code), staff\_no (p\_staff\_no) and quantity (p\_quantity) as parameters.
   2. Declare two local variables, one to hold quantity in stock (v\_quantityinstock) and one to hold number of staff (v\_num\_staff).
   3. Select the count of staff members that have the staff\_no p\_staff\_no from the staff table into v\_num\_staff.
   4. If this number is zero, raise an exception 'no such staff member'. Otherwise, continue.
   5. Select the quantityinstock for the stock\_code p\_stock\_code from sh\_stock and store it in v\_quantityinstock.
   6. If v\_quantityinstock is less than p\_quantity, rais an exception 'Not enough stock'. Otherwise continue.
   7. Insert a row into sh\_sale, using the parameters passed, with now() as the saledate.
   8. Update the sh\_stock row for the stock code with value p\_stock\_code to reduce quantityinstock by the amount sold.
   9. Add standard exception handling to the end of the procedure.
   10. Test your procedure for robustness with valid and invalid parameter values.
3. Set up AddSupplier.
   1. Upload 'AddSupplier.sql'.
   2. Amend the code for use in your own schema (i.e. change "CornerShop" to your schema).
   3. Working with the person next to you: Find out their student number and grant usage to your schema to their role.
   4. Grant the person next to you sufficient privileges to allow them to add a supplier to your table, using your addSupplier function.
4. Write a trigger to audit the addition of suppliers:
   1. Create a supplier\_log table to hold supplier\_id, supplier name, supplier\_id, username who added the supplier and time the supplier was added.
   2. Create a trigger audit\_supplier to record any addition of suppliers that has happened.
   3. Test the trigger using the function AddSupplier.

**SEE RUBRIC:**

* 1 mark for getting your procedure fully working
* Half mark for getting your trigger working
* Half mark for getting your trigger working when your neighbour adds a supplier, using your function and firing the 'audit\_supplier' trigger.

Description of Corner Shop: A local business called the Corner Shop acts as a newsagent and general shop to the public. There are a few staff members working there. Each staff member is allocated a unique staff\_no when they join the shop and their staff no is recorded, with their name, in the SH\_STAFF table. There is a wide variety of stock and most of the stock comes from recorded suppliers. Stock is given a category, so that it can be properly stored. When stock comes into the shop, if it is not stock that was sold previously, it is given a unique stock code. The stock\_code, stock\_name and price for each stock item is recorded in the STOCK table, along with the quantity (quantityinstock). It is given a CatCode to show its category from the SH\_CATEGORY. Category descriptions include 'Stationery', 'Long-life food', 'refrigerated food' and 'heavy items'. If the stock came in from a regular supplier, that supplier’s supplier\_id is recorded in the SH\_STOCK table. However, sometimes the shop sells items such as homemade cakes, or flags or bunting for some event. These do not have a supplier\_id. Regular suppliers have a unique supplier\_id, a name (sname) and a phone number (sph) that are recorded in the SH\_SUPPLIER table. When a sale is made, the stock\_code and quantity is recorded in the SH\_SALE table, along with the STAFF\_NO of the staff member who made the sale, and the date and time of the sale.

TABLES:

* SH\_SUPPLIER (SUPPLIER\_ID(PK), SNAME, SPH)
* SH\_STAFF(STAFF\_NO(PK), STAFF\_NAME)
* SH\_CATEGORY (CatCode(PK), CatDescription, StorageConditions)
* SH\_STOCK(STOCK\_CODE(PK), SUPPLIER\_ID(FK optional), STOCK\_NAME, QUANTITYINSTOCK, PRICE, CatCode)
* SH\_SALE(STOCK\_CODE(PK, FK), STAFF\_NO(PK, FK), SALEDATE(PK), QUANTITY)

RELATIONSHIPS:

* SH\_SUPPLIER: SH\_STOCK 0..1:0..many using SUPPLIER\_ID , non-identifying
* SH\_CATEGORY:SH\_STOCK 1:0..many using CatCode, non-identifying
* SH\_STAFF: SH\_SALE 1:0..many using STAFF\_NO, identifying
* SH\_STOCK: SH\_SALE 1:0..many using STOCK\_CODE, identifying

NOTE: An identifying relationship means that the foreign key also acts as part of the primary key. In this case, the SH\_SALE primary key is a combination of STOCK\_CODE, STAFF\_NO and SALEDATE. SALEDATE is used so that the same staff member can sell the same stock item multiple times.