STORING AND RETRIVING DATA - FINAL PROJECT

Group 36

Members: 2

Marina Lashina 20220728

Natalia Puzina 20220722

Course:
[7512] Master in Data Science and Advanced Analytics

Fall semester 2022/2023

Commercial business process description

Getpresent is an online service allowing its customers to greet their friends, relatives, colleagues or “significant others” no matter how distant they are. The customer just has to spend few minutes visiting company´s website, select the products from the list and provide the detailed address for delivery to initiate the order.

Getpresent operates with 3 product categories:

- flowers
- wine
- cakes

The concept assumed to satisfy different price segments, so the customers can choose products just from 1 category or any combinations above as well as have certain variety of product prices.

The company is located in Portugal, but it serves clients from both Europe and overseas with the target delivery in Europe.

The company introduced its clients to a loyalty programs allowing certain discount rates to be applied depending on customer´s loyalty program status.

Getpresent has a concept of maintaining a relationship with the wide range of individuals and small entrepreneurship suppliers to provide exclusive and “price-convincing” products to its customers. It´s very important for a company to receive the regular feedback from customers and their satisfaction with both purchased products and quality of delivery services, so the company uses rating system to monitor quality of services and products.

Getpresent recently provides its services to individuals only, however it has a plan to expand its services for corporate customers as well. The launch of corporate customer services requires substantial upgrade of internal control system.

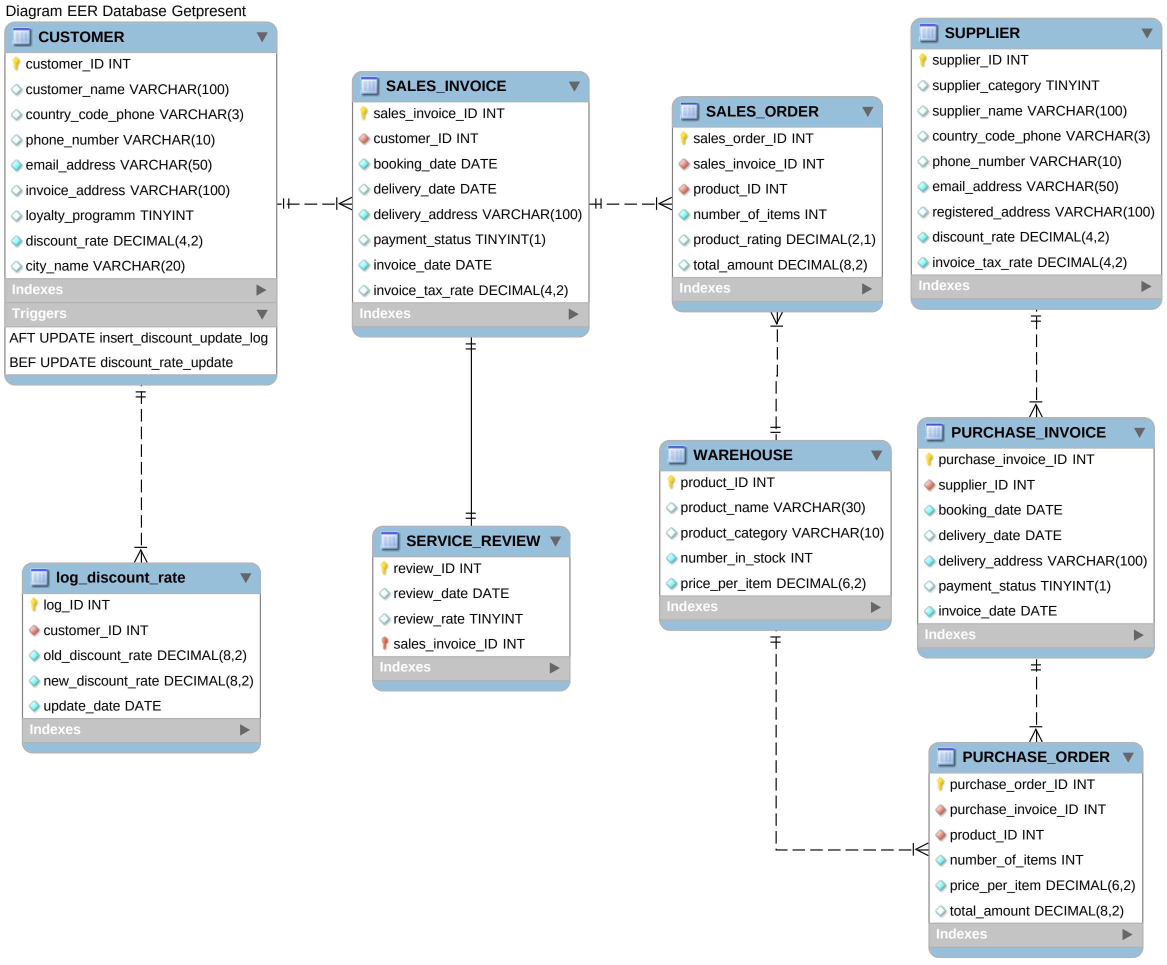
While working with SQL database development consultants, company´s management requested to database design to be as efficient as possible – in order to achieve max simplicity for users on both input, update and analyse information stored and minimize the potential number of user’s errors.

Company´s employment policy is to assign the vacancies to for part-timers, such as students or retired people, so one of the expectations from database design was to achieve short training time required for those with lack of advanced previous experience using databases or any special software packages.

DATABASE SET UP AND DESIGN

ERD of database named Getpresent using 9 tables covering the part of Company´s business transactions is presented below.

Diagram EER Database Getpresent



Database was created and designed to ensure all three normal formed are complied with:

- all attribute types are atomic and single-valued, no multivalued attribute types are used in order to comply with 1st normal form;

- assuming 1st normal form complied with, when all of its non-key attributes are fully functionally dependent on its primary key in order to comply with 2nd normal form;

- assuming 2nd normal form complied with, no non-key attribute is transitively dependent on primary key in order to comply with 3rd normal form.

9 tables were created to support the following base activities:

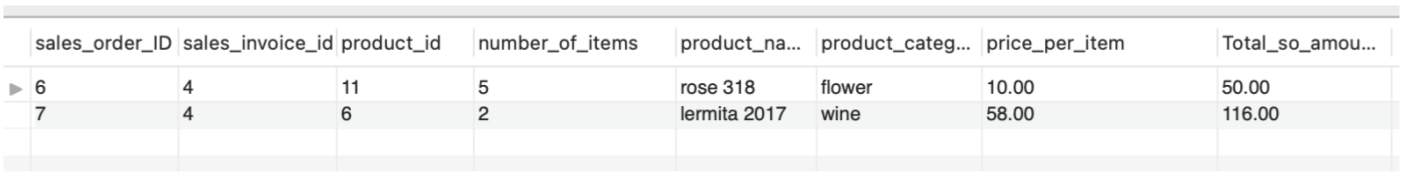
- **Customer** and **Supplier** tables, storing the max complete set of information required in order to communicate and maintain necessary administrative, accounting and transactional documents turnover;
- **Sales invoice** and **Sales order** tables were designed to support customer´s orders initiation and relevant accounting document (invoice) producing for proper reflection of company´s sales activities;
- **Purchase invoice** and **Purchase order** tables were designed to support company´s purchasing activity and relevant accounting document (invoice) producing for proper reflection of company´s purchasing activities;
- **Warehouse** table, being a “connection point” of company´s sales and purchases, storing the necessary information about products available, its pricing and number of products readily available for sale;
- **Service rating** table was created to support company´s efforts to receive a customer´s feedback on its services based on their experience ordering from a company;
- **log_discount_rate** table was created to implement trigger log function.

We´ve generated the input data for the tables necessary to perform project tasks. The data was partially taken from our training database used for SRD practical class (HR database was used to generate data for input into Customer table), the other data (values used for Warehouse, Sales order, Sales invoice) were generated from open sources with the manual input into relevant tables.

In order to achieve the max efficiency, we were avoiding creating duplication of attributes within different tables, instead we were developing some “views”, allowing us to aggregate attributes from different tables and select relevant data from it.

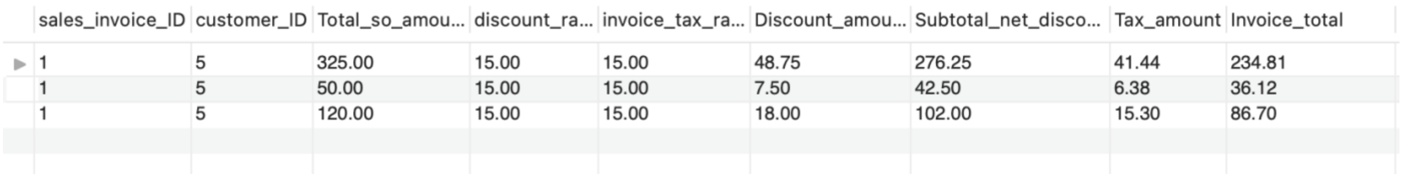
For example,

1. View **sorderdetailed** aggregates more detailed information on the Sales order from 3 different tables with the total amount of Sales order being also calculated in it. To demonstrate how view works we’ve selected sales_invoice_ID 4 that consists of 2 sales_orders, 6 and 7.



sales_order_ID	sales_invoice_id	product_id	number_of_items	product_na...	product_categ...	price_per_item	Total_so_amou...
6	4	11	5	rose 318	flower	10.00	50.00
7	4	6	2	lermita 2017	wine	58.00	116.00

2. View **sinvoicefin aggregates** information from 2 different tables plus from view **sorderdetailed** for calculations of summary financial number required for sales_invoice generation. To demonstrate how view works we’ve selected sales_invoice_ID 1.



sales_invoice_ID	customer_ID	Total_so_amou...	discount_ra...	invoice_tax_ra...	Discount_amou...	Subtotal_net_disco...	Tax_amount	Invoice_total
1	5	325.00	15.00	15.00	48.75	276.25	41.44	234.81
1	5	50.00	15.00	15.00	7.50	42.50	6.38	36.12
1	5	120.00	15.00	15.00	18.00	102.00	15.30	86.70

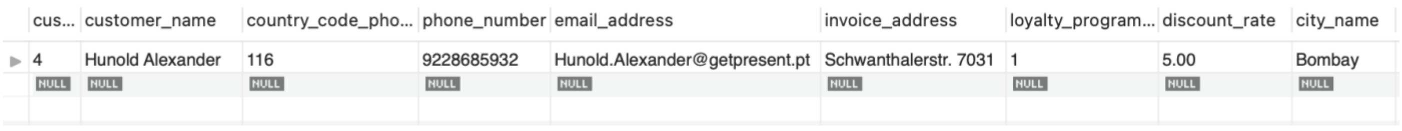
In order to minimize the potential input errors, we´ve tried to make max “tailoring” of the datatypes used for a different attribute.

For example:

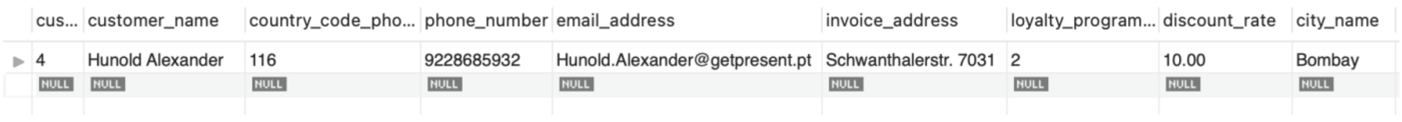
- assuming our company do not sell the goods with the sale and purchase prices equal or over 10,000 Euro – we´ve assigned the datatype DECIMAL (6,2) to be used to reflect “financial” data of 6 digits (including 2 “reserved” for eurocents), so no value exceeding 9,999.00 allowed on input;
- though both purchase and customer order could potentially create for amount exceeding 10,000 Euro – we´ve assigned datatypes DECIMAL (8,2) for those;
- on set up of financially specific rates, such as discount rate or invoice tax rate, datatypes DECIMAL (4,2) – as it is assumed by neither tax no discount rates could go over 4 digits (including 2 decimals);
- we used TINYINT datatype to deal with loyalty programs (assuming there is no substantial diversification of those used by the company),
- we´ve assigned DECIMAL (2,1) for product and service ratings, assuming the rating system from 1 to 5 with 1 decimal possible etc

The following triggers were created to perform task C:

1st trigger for UPDATE. Trigger automatically updates of discount rate in the table **customer** if level of loyalty program changes.

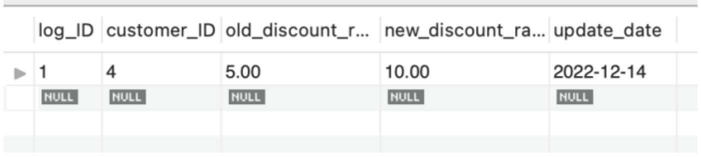


cus...	customer_name	country_code_pho...	phone_number	email_address	invoice_address	loyalty_program...	discount_rate	city_name
4	Hunold Alexander	116	9228685932	Hunold.Alexander@getpresent.pt	Schwanthalerstr. 7031	1	5.00	Bombay
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL



cus...	customer_name	country_code_pho...	phone_number	email_address	invoice_address	loyalty_program...	discount_rate	city_name
4	Hunold Alexander	116	9228685932	Hunold.Alexander@getpresent.pt	Schwanthalerstr. 7031	2	10.00	Bombay
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

2nd trigger inserts a row in log table **log_discount_rate.** This table fills automatically when a discount rate of customer changes.



log_ID	customer_ID	old_discount_r...	new_discount_ra...	update_date
1	4	5.00	10.00	2022-12-14
NULL	NULL	NULL	NULL	NULL

In the task F2 we show the list of the 3 most popular products which were ordered the most frequently.
In the task F5 we show the list all the locations (city and country from delivery address) where products were sold, and the product has customer’s ratings.

All tables and views required for tasks are forming after running query.

Addendums to the above report:

1. **SQL script “Database and data input”**
2. **SQL script “Task F”**
3. **SQL script “INVOICE” (Task G)**
4. **SQL script “Demonstration of triggers”**