IST 302 G5 Database Project

Eman Alzahrani, Daniel Perez, Shengyu Zhang, Aote Xin

Executive Summary

The IST 302 Database Project aims to develop a sales transaction application with a focus on database design and enhancement procedures. The project is organized into several sections, each addressing specific aspects of the database development process.

All the scripts are shared on a github repository (<https://github.com/axin-geo/IST302DBG5FINAL)>

# 

# **Table of Contents**

[Table of Contents 2](#_nlgsxatfr8t3)

[Section 1: Business Understanding 3](#_37wfqj13jjav)

[Overview of sales transaction applications 3](#_3102yolw1c6u)

[List of users for the sales transaction system 3](#_2mbwmtmtw78a)

[Users and their business queries, including its description, the required database table access, and access condition 3](#_rgkxuwtq95ph)

[Section 2: Data Understanding 5](#_ymw6juluwoqu)

[Description of all data understanding tasks performed and their outputs 5](#_ry6mvbpvay0a)

[Section 3: Conceptual Design (Shengyu) 12](#_4qpase3yfyxn)

[Present the conceptual model (ERD) of the database system, including the justifications 12](#_5h51e04nlad0)

[Section 4: Security Requirements (Shengyu) 13](#_vh613ywd5wou)

[Present initial access permission interaction matrix (select, update, insert, delete) 13](#_v9xlw7ayoech)

[List the business views that will be implemented as stored procedures 13](#_5ftv2vnix9xz)

[Section 5: Logical Design 15](#_4cjoba68d72e)

[Present the logic mode (database diagram) of the database system. 15](#_o80lqhr4obuv)

[Specify constraints of the database system 15](#_hx0scsgmsanf)

[Section 6: Physical Design (Daniel) 18](#_epy0quvk8d3b)

[Present the physical data model, including DDL script to create tables with constraints 18](#_pvqltdsutu8)

[Scripts to create, set constraints and fix data from the tables: 18](#_rtcy37z9tflj)

[Section 7: Database Implementation (AX) 21](#_u9rs87tkfw95)

[DML scripts to populate tables 21](#_ywuw1qi5uyhw)

[Test integrity constraints 23](#_1mdtsirv23sf)

[Design and implement data access structure for the database system, including indexes, views, and stored procedures 24](#_hipwf1lr8j3d)

[Section 8: Conclusion 29](#_h2iy7qlpeyjc)

# 

# 

# Section 1: Business Understanding

## Overview of sales transaction applications

A sale is an exchange of a product (or service) between two or more parties in exchange of money or some other compensation. It takes place almost every day in human-life. Whenever there is a sale, there are at least two parties involved; a seller who is typically an owner of the product or who has legal possession and a buyer who is seeking the product.

A transaction is defined as a piece of business, such as the act of buying or selling something. An application for sale transactions is the system that allows a customer to process sales transactions, usually in exchange of cash or an account receivable.

Some examples for such applications could be the software used in the supermarkets where the sales transactions get registered in the company’s database to be processed and accessed by diverse users. Another example we can think of is the software company that sells products online and they have a platform where they process all the sales transactions for their software, this software is used by multiple teams in the company.

## List of users for the sales transaction system

There are many users we identified that could be using the sales transaction system, among those we will find the developers, managers, accountants, etc. For purposes of this project we are going to focus on 2 groups of users, the sales department and the marketing department.

Sales users need a transactional database system to maintain sales history, keep track of customer information, shipping, budgets and product information among others.

Marketing users apply the transaction database system to create marketing and promotion strategies, keep track of product sales and customer behaviors, or to check certain sales channels for example.

## Users and their business queries, including its description, the required database table access, and access condition

**Business Query 1:** Select any channel from the G5\_CHANNELS that was NOT used by any sale transaction made.) This query provides information on channels not being used, opening possibilities to explore potential issues with sales channels or sales strategies. It could be useful information for the sales and marketing departments.

This query relies on access to the sales table and provides reading only privileges since it is not necessary to update or delete transactions.

**Business Query 2:** Search sales transactions history for a given date. This query can be made by sales and marketing departments using the business view 2 (transactions for 2012) in which we join products and sales tables conditioned to transactions happening over 2012.

**Business Query 3:** This query provides information about Internet sales promotion history. Providing the transactions made over the internet channel over a certain period of time. This query accesses the Business view 3, where we query sales, channels and promotions tables and filter out sales made over the internet within a determined period of time in 2012.

This particular query will be used for the marketing department, and will be a read only query because for analysis we don’t need to modify any data in the tables.

**Business Query 4:** 2012 Internet Sales Promotion History in Asia: This query is for promotions on sales that happened during 2012 via the internet channel in the continent of Asia. It will be used by the marketing department to analyze how the sales went in the Asian market during 2012. They should have read only permissions because it is not needed to modify the data in the tables using the Business view 4.

**Business Query 5:** Customer demography. This query provides information on the gender and their total price over internet purchases during 2021.

This query is made by the marketing department and read only access is necessary over the business view 5.

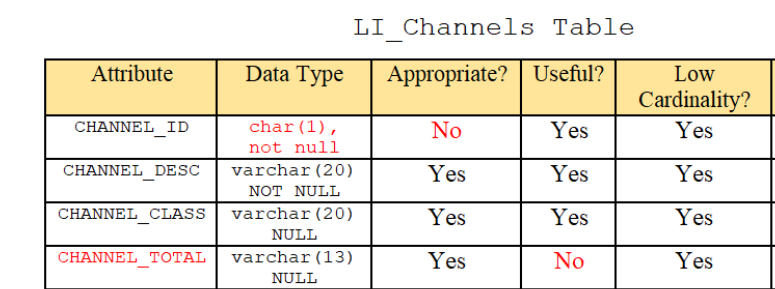
**Business Query 6:** calculate the total sales made in a continent in 2012 via the internet. This query is used by the marketing and sales department to verify the total sales categorized by continents. The sales and marketing department should have read only access to the table channels, customers, sales and country to do this query.

**Business Query 7:** Select the product that has the highest sale in 2012 in a continent via a certain channel. This query is used by the marketing and sales department to verify the product with highest sales categorized by continents. The sales and marketing department should have read only access to the table channels, customers, sales and country to do this query.

While in general the queries mentioned don't need modify or delete privileges the sales department and marketing should have those privileges on the products table. Marketing should have those on the promotions and customers table as well, and the sales should have those on the channels and countries table.

# Section 2: Data Understanding

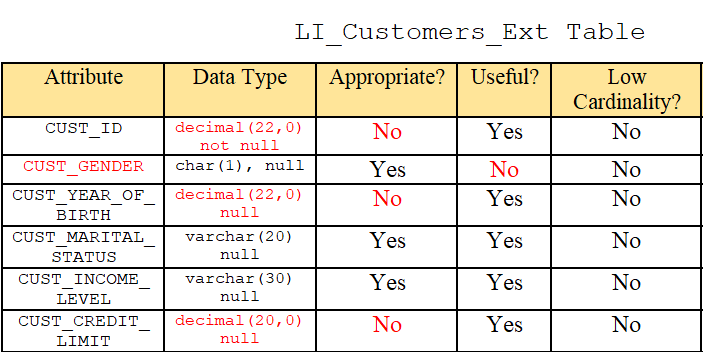
## Description of all data understanding tasks performed and their outputs



⦁ We deemed the data type for CHANNEL\_ID as inappropriate due the fact that the char(n) data type stores character data. With CHANNEL\_ID being a primary key, the INT data type is the most appropriate.

⦁ CHANNEL\_TOTAL is a useless column, as its values repeat and do not hold significant information for the users.

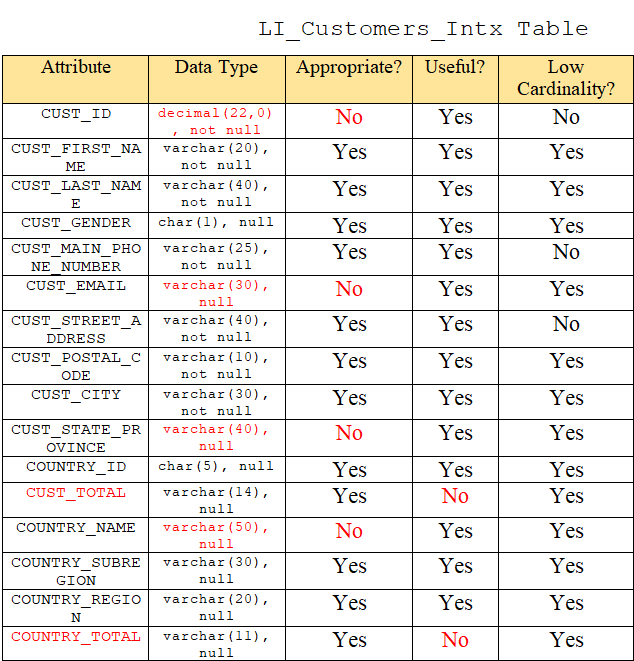
⦁ Each attribute has low cardinality, due to the limited number of values populated in it.



⦁ We deemed the data type for CUST\_ID as inappropriate due to the fact that the decimal(p,s) data type is more suited towards values with decimals rather than integers. We replaced this with INT, as most primary keys have this data type, as well.

⦁ We found CUST\_GENDER to be a useless column, as there is already a column that indicates customer gender on the LI\_CUSTOMERS\_INTX table.

⦁ The decimal(p,s) data types for both CUST\_YEAR\_OF\_BIRTH and CUST\_CREDIT\_LIMIT are deemed inappropriate since both attributes contain values that are integers and not real numbers.



⦁ The data type for CUST\_ID is deemed inappropriate since the values of the attribute contain integers instead of real numbers.

⦁ For CUST\_EMAIL’s data type, although varchar(n) was the appropriate data type, we decided to decrease the n value to 50 rather than 30.

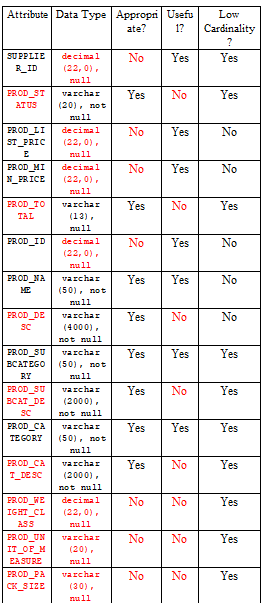
⦁ Regarding CUST\_STATE\_PROVINCE’s data type, we did deem varchar(40) as appropriate; however, instead of NULL, we replaced it with NOT NULL. Because the customer’s state is necessary in order for the post office to mail the products to the customer successfully.

⦁ We deemed the CUST\_TOTAL attribute as useless due to the fact that the data repeats and holds no significant value to the users.

⦁ With COUNTRY\_NAME’s data type, we did find that varchar(50) was appropriate; however, having the country data is necessary in order to properly mail products to the customers. Therefore, rather than NULL, we changed it to NOT NULL.

⦁ We deemed COUNTRY\_TOTAL as an useless column since all the values repeat and hold no significant value to the user.

LI\_Products Table

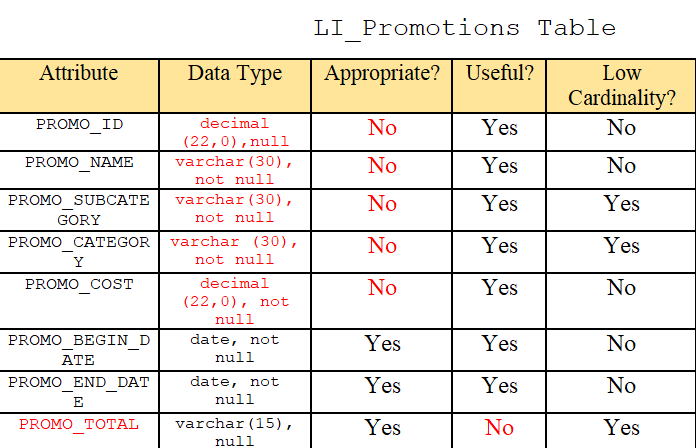


⦁ Regarding the decimal(p,s) datatype of the SUPPLIER\_ID and PROD\_ID attributes, such is deemed as inappropriate since the values of both attributes are integers and not real numbers. PROD\_ID serves as the primary key of this table.

⦁ The attributes PROD\_STATUS, PROD\_TOTAL, PROD\_WEIGHT\_CLASS, PROD\_UNIT\_OF\_MEASURE, and PROD\_PACK\_SIZE are deemed as useless since all the columns repeat and hold no value to the user in terms of decision-making.

⦁ The varchar(n) data type for PROD\_UNIT\_OF\_MEASURE and PROD\_PACK\_SIZE are deemed inappropriate since all the values contain one character only, so CHAR(1) would have been more appropriate.

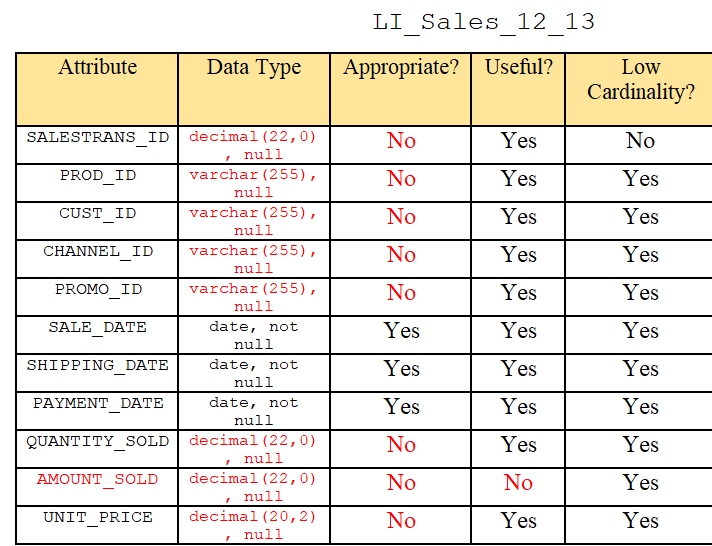
⦁ We deemed PROD\_DESC, PROD\_SUBCAT\_DESC and PROD\_CAT\_DESC as useless because there are other attributes that hold the exact values as all these particular columns, which causes a sense of redundancy.



⦁ The PROMO\_ID data type is inappropriate since this attribute serves as a primary key, and INTs are more suitable data types for primary keys. In addition, the data in PROMO\_ID contains integers and not real numbers.

⦁ We deemed the decimal(22,0) data type for PROMO\_COST to be inappropriate.

⦁ The PROMO\_TOTAL attribute is considered useless, as the data does not hold valuable insight to the users and has repeating values per each column.

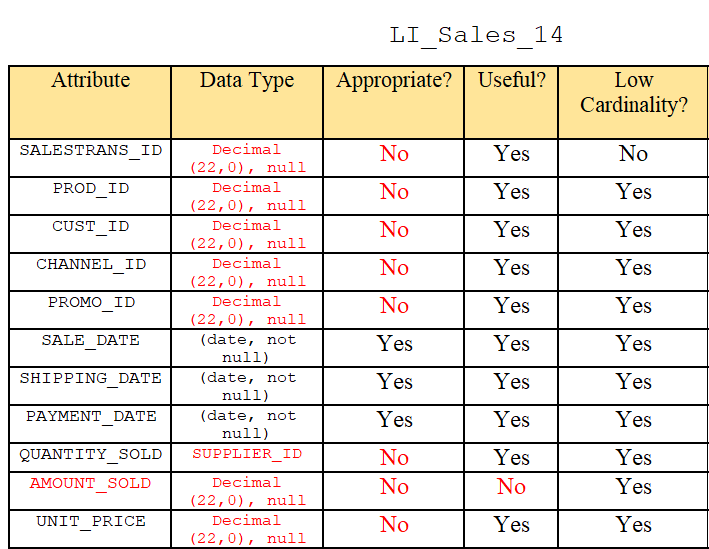


⦁ The SALESTRANS\_ID data type is inappropriate since this attribute serves as a primary key, and INTs are more suitable data types for primary keys. In addition, the data in SALESTRANS\_ID contain integers and not real numbers.

⦁ The varchar(255) data type is inappropriate for PROD\_ID, CUST\_ID, CHANNEL\_ID, and PROMO\_ID since all of these columns serve as primary keys for other tables and have numeric values of integers. The appropriate data type would be INT.

⦁ The decimal(p,s) data type for QUANTITY\_SOLD and AMOUNT\_SOLD is inappropriate since the values of both attributes are integers and not real numbers.

⦁ The AMOUNT\_SOLD attribute is useless, since these values are identical to the values in the UNIT\_PRICE column.



⦁ The SALESTRANS\_ID data type is inappropriate since this attribute serves as a primary key, and INTs are more suitable data types for primary keys. In addition, the data in SALESTRANS\_ID contain integers and not real numbers.

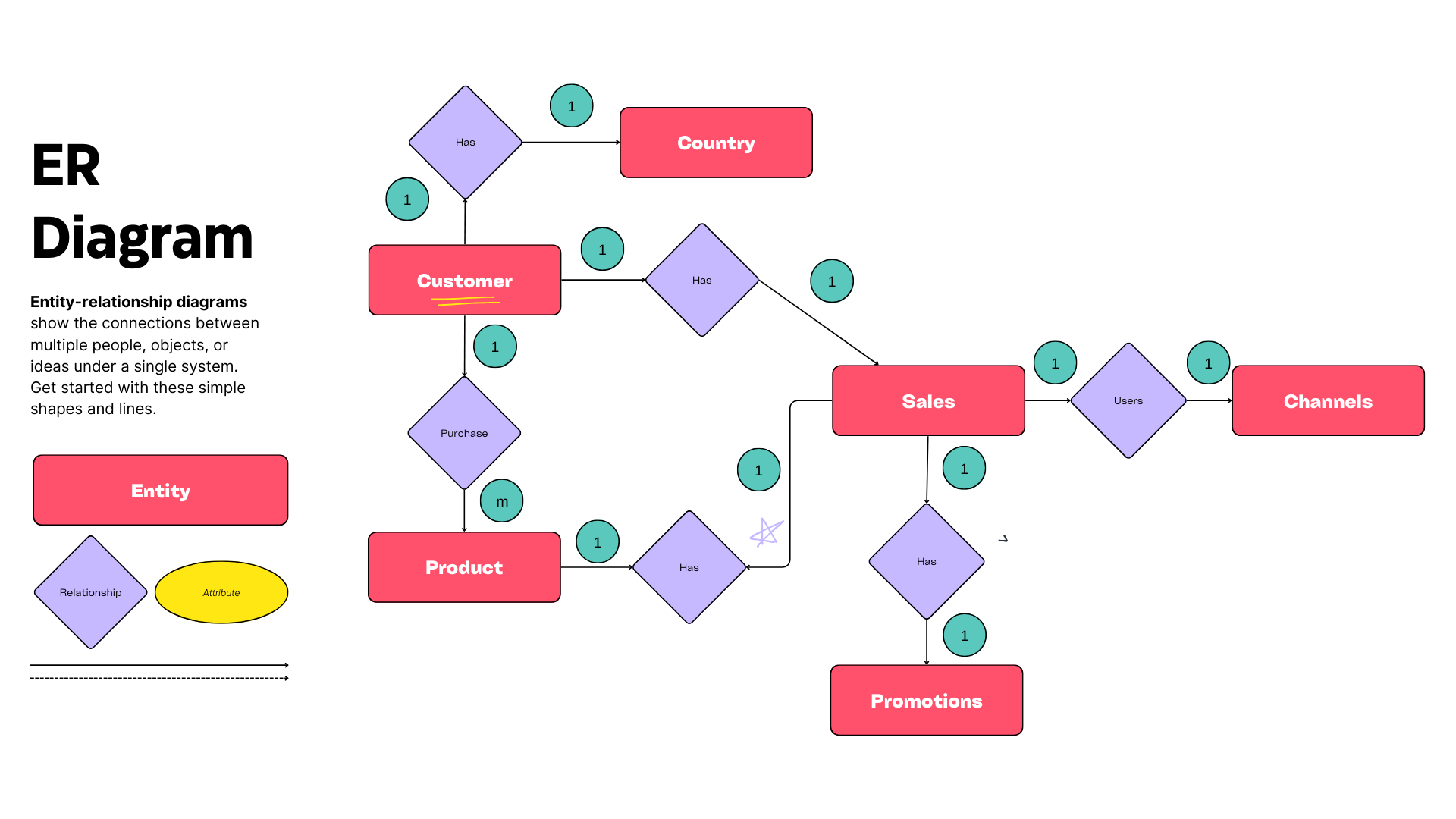
⦁ The varchar(255) data type is inappropriate for PROD\_ID, CUST\_ID, CHANNEL\_ID, and PROMO\_ID since all of these columns serve as primary keys for other tables and have numeric values of integers. The appropriate data type would be INT.

⦁ The decimal(p,s) data type for QUANTITY\_SOLD and AMOUNT\_SOLD is inappropriate since the values of both attributes are integers and not real numbers.

⦁ The AMOUNT\_SOLD attribute is useless, since these values are identical to the values in the UNIT\_PRICE column.

# Section 3: Conceptual Design

## Present the conceptual model (ERD) of the database system, including the justifications



This is an Entity-Relationship (ER) diagram that illustrates the relationships between different entities in a database system.

Here's a description of the components and their connections:

Entities: There are five rectangular entities labeled 'Customer', 'Country', 'Sales', 'Channels', and 'Promotions'.

Relationships: The entities are connected by diamonds indicating relationships. These are labeled 'Has' for the connections between 'Country' and 'Customer', 'Customer' and 'Sales', 'Sales' and 'Channels', and 'Sales' and 'Promotions'.

Attributes: The oval shapes represent the attributes of entities or relationships. In this diagram, the 'Customer' entity has an attribute labeled 'Purchase'.

Cardinality: The connections have cardinality indicators such as '1' or 'm', signifying one-to-one or one-to-many relationships. For example, a '1' near 'Country' and a 'm' near 'Customer' implies that one country can have many customers, but each customer is associated with only one country.

Legend: At the bottom left, there is a legend explaining that rectangles represent entities, diamonds represent relationships, ovals represent attributes, and the lines connecting these shapes illustrate the links between them.

# Section 4: Security Requirements

## Present initial access permission interaction matrix *(select, update, insert, delete)*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***BUSINESS VIEWs*** | | | | |
| ***USERs*** | BV02 | BV03 | BV04 | BV05 |
| Sales (sale) | S |  | ISUD | ISUD |
| Marketing (mark) |  | ISUD | IS | ISUD |

I: Insert

S: Select

U: Update

D: Delete

## List the business views that will be implemented as stored procedures

The business views that will be implemented as stored procedures are:

**BV02:** Sales Transaction History (Join Products table and Sales table). This table focuses on querying for the sales transactions that happened during a particular date/dates. Sales Product Performance evaluation.

**BV03:** Sales Promotion History: (Join Sales table and Promotions table). This table focus on query for sales promotions that happened during a particular date/dates.

**BV04:** Sales Channel (Join Channel and Sales Table). To know via which channel we have the highest sale for each product.

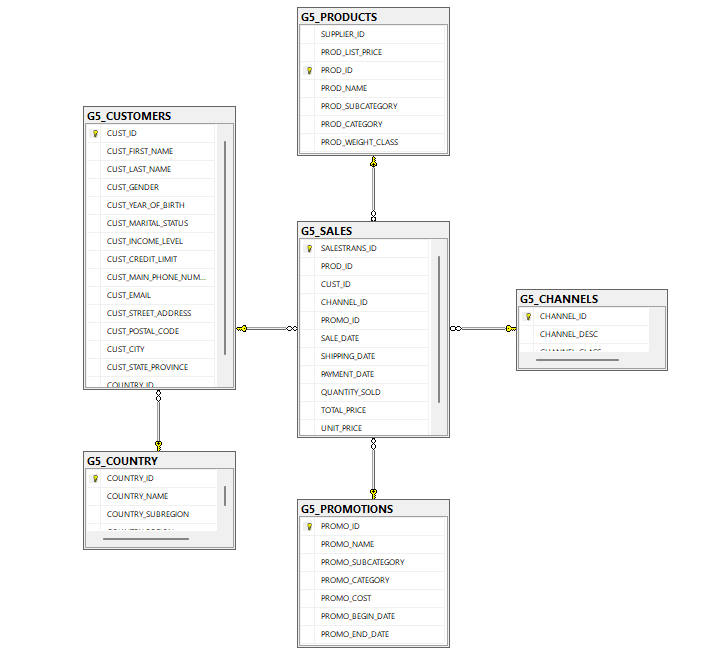
**BV05**: Customer Demography (Join external and internal customers table together). Answer questions including 1. from which age group of our customers have the highest sale 2. from which gender group of our customers have the highest sale.

# Section 5: Logical Design

## Present the logic mode (database diagram) of the database system.

These are the step we design this logical model:

1. Database Analysis: We initiate the process by conducting a comprehensive analysis of the existing database. This involves scrutinizing each piece of data to differentiate between critical and superfluous information.
2. Determining Key Features: Once we understand the data landscape, we identify the key features and information that the database must retain. These are the pieces of data that are essential for the operations and functionality the database supports.
3. Data Cleaning: With the key features earmarked, we proceed to purge the database of any redundant or irrelevant data. This step ensures that the database is not cluttered with unnecessary information that could hamper efficiency or clarity.
4. Redesign and Consolidation: The next phase is the redesign of the database structure. This involves merging related data to reduce repetition and fragmentation. The consolidation process is aimed at achieving a more streamlined and logical data organization.
5. Structural Integrity: In parallel with consolidation, we meticulously define and verify the primary and foreign keys. Ensuring that primary keys are unique and that foreign keys correctly establish the relationships between tables is crucial for maintaining the integrity and consistency of the database.
6. Normalization: We apply normalization rules to organize the data in the database. Specifically, we notice that a few country and region related columns do not have dependency on the main part of the customers table. Therefore we created a new table named G5\_COUNTY. This step minimizes redundancy and dependency by dividing the database into tables and establishing relationships between them.



## Specify constraints of the database system

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Database Tables Table Name** | **Primary Key Constraint** | **Referential integrity constraints** | **Relevant inter-columns constraints** | **Domain Constraints** |
| G5\_CHANNELS | CHANNEL\_ID |  | **CHANNEL\_ID**, : int, notNULL, Unique  CHANNEL\_DESC: Varchar, notNULL  CHANNEL\_CLASS: Varchar, notNull |  |
| G5\_CUSTOMERS | CUST\_ID |  | **CUST\_ID** : int notNull  CUST\_FIRST\_NAME Varchar(40) CUST\_LAST\_NAME Varchar(40)  CUST\_GENDER char(1) CUST\_MAIN\_PHONE\_NUMBER varchar(25)  CUST\_EMAIL varchar(30) CUST\_STREET\_ADDRESS varchar(30)  CUST\_POSTAL\_CODE varchar(10)  CUST\_CITY varchar(30)  CUST\_STATE\_PROVINCE CUST\_YEAR\_OF\_BIRTH int(4)  CUST\_MARITAL\_STATUS varchar(20) CUST\_INCOME\_LEVEL varchar(30) CUST\_CREDIT\_LIMIT decimal(20, 0) | 1. *[CUST\_GENDER] (M,F),* 2. *[CUST\_MARITAL\_STATUS] (Single, married, divorced, null, widowed, separate)* |
| G5\_PRODUCTS | PROD\_ID |  | **PROD\_ID**: int, notNULL  PROD\_NAME: Varchar, notNULL  PROD\_SUBCATEGORY: Varchar notNULL  PROD\_CATEGORY:Varchar, notNULL  PROD\_WEIGHT\_CLASS: int notNULL PROD\_LIST\_PRICE: decimal, notNULL |  |
| G5\_PROMOTIONS | PROMO\_ID |  | **PROMO\_ID** (integer, not null),  PROMO\_NAME (varchar(30), not null),  PROMO\_SUBCATEGORY (varchar(30), not null),  PROMO\_CATEGORY (varchar(30), not null),  PROMO\_COST (integer, not null),  PROMO\_BEGIN\_DATE (date, null), PROMO\_END\_DATE (date, null) |  |
| G5\_SALES | SALESTRANS\_ID | CUST\_ID  PROD\_ID  PROMO\_ID  CHANNEL\_ID | **SALESTRANS\_ID (integer, not null, unique),**  PROD\_ID (integer, not null),  CUST\_ID (integer, not null),  CHANNEL\_ID (integer, not null),  PROMO\_ID (integer, not null),  SALE\_DATE (date, null),  SHIPPING\_DATE (date, null),  PAYMENT\_DATE(date, null),  QUANTITY\_SOLD (integer, not null),  AMOUNT\_SOLD(decimal, not null),  UNIT\_PRICE(decimal, not null) |  |
| G5\_COUNTRY | COUNTRY\_ID |  | COUNTRY\_ID char(5)  COUNTRY\_NAME varchar(50)  COUNTRY\_SUBREGION varchar(30)  COUNTRY\_REGION varchar(20) | *[COUNTRY\_REGION] (Oceania, Europe, Americas, Africa, Asia)* |

# Section 6: Physical Design

## Present the physical data model, including DDL script to create tables with constraints

## 

## Scripts to create, set constraints and fix data from the tables:

|  |
| --- |
| -- 1. G5\_CHANNELS Table  --- 1.1 create table  CREATE TABLE G5\_CHANNELS (  CHANNEL\_ID int not NULL unique,  CHANNEL\_DESC varchar(50) not NULL,  CHANNEL\_CLASS Varchar(50) not Null  CONSTRAINT PK\_CHANNEL\_ID PRIMARY KEY clustered (CHANNEL\_ID)  )  -- 2. Create G5\_Customer Table with the data field definition we want  --- 2.1 Create G5\_CusEx table  select \* into G5\_CusEx  from [dbo].[LI\_CUSTOMERS\_EXT]  --- 2.2 Create G5\_CusIn table  select \* into G5\_CusIn  from [dbo].[LI\_CUSTOMERS\_INTX]  --- 2.3 Join tables together  select cx.[CUST\_ID], [CUST\_FIRST\_NAME],[CUST\_LAST\_NAME],  cx.[CUST\_GENDER],[CUST\_YEAR\_OF\_BIRTH],[CUST\_MARITAL\_STATUS],  [CUST\_INCOME\_LEVEL],[CUST\_CREDIT\_LIMIT],  [CUST\_MAIN\_PHONE\_NUMBER],[CUST\_EMAIL],  [CUST\_STREET\_ADDRESS],[CUST\_POSTAL\_CODE],  [CUST\_CITY],[CUST\_STATE\_PROVINCE],  [COUNTRY\_ID],[CUST\_TOTAL],  [COUNTRY\_NAME],[COUNTRY\_SUBREGION],  [COUNTRY\_REGION],[COUNTRY\_TOTAL]  into G5\_CUSTOMERS  from G5\_CusEx cx  join G5\_CusIn ci on cx.CUST\_ID = ci.CUST\_ID  --- 2.5 alter table G5\_Customer table  alter table G5\_CUSTOMERS alter column CUST\_ID int not Null;  alter table G5\_CUSTOMERS alter column CUST\_FIRST\_NAME Varchar(50);  alter table G5\_CUSTOMERS alter column CUST\_LAST\_NAME Varchar(40);  alter table G5\_CUSTOMERS alter column CUST\_GENDER char(1);  alter table G5\_CUSTOMERS alter column CUST\_MAIN\_PHONE\_NUMBER varchar(25);  alter table G5\_CUSTOMERS alter column CUST\_EMAIL varchar(30);  alter table G5\_CUSTOMERS alter column CUST\_STREET\_ADDRESS varchar(40);  alter table G5\_CUSTOMERS alter column CUST\_POSTAL\_CODE varchar(10);  alter table G5\_CUSTOMERS alter column CUST\_CITY varchar(30);  alter table G5\_CUSTOMERS alter column CUST\_STATE\_PROVINCE varchar(40);  alter table G5\_CUSTOMERS alter column COUNTRY\_ID char(5);  alter table G5\_CUSTOMERS alter column COUNTRY\_NAME varchar(50);  alter table G5\_CUSTOMERS alter column COUNTRY\_SUBREGION varchar(30);  alter table G5\_CUSTOMERS alter column COUNTRY\_REGION varchar(20);  alter table G5\_CUSTOMERS alter column CUST\_YEAR\_OF\_BIRTH int;  alter table G5\_CUSTOMERS alter column CUST\_MARITAL\_STATUS varchar(20) null;  alter table G5\_CUSTOMERS alter column CUST\_INCOME\_LEVEL varchar(30);  alter table G5\_CUSTOMERS alter column CUST\_CREDIT\_LIMIT decimal(20, 0);  alter table G5\_CUSTOMERS drop column COUNTRY\_TOTAL;  alter table G5\_CUSTOMERS drop column CUST\_TOTAL;  alter table G5\_CUSTOMERS ADD CONSTRAINT PK\_CUST\_ID PRIMARY KEY clustered (CUST\_ID);  alter table G5\_CUSTOMERS ADD CONSTRAINT CONSTRAINT\_GENDER CHECK (CUST\_GENDER='f' OR CUST\_GENDER='m' OR CUST\_GENDER='F' OR CUST\_GENDER='M');  alter table G5\_CUSTOMERS ADD CONSTRAINT CONSTRAINT\_MARITAL\_STATUS CHECK (CUST\_MARITAL\_STATUS='single' OR CUST\_MARITAL\_STATUS='married' OR CUST\_MARITAL\_STATUS='divorced'OR CUST\_MARITAL\_STATUS='separated' OR CUST\_MARITAL\_STATUS='widowed' OR CUST\_MARITAL\_STATUS is null);  -- 3. Create Country table from G5\_CUSTOMERS  go  select DISTINCT [COUNTRY\_ID], COUNTRY\_NAME, COUNTRY\_SUBREGION, COUNTRY\_REGION  into G5\_COUNTRY  from G5\_CUSTOMERS  Go  – 3.1 Alter table constraints and column settings  alter table G5\_COUNTRY alter column COUNTRY\_ID int not null  go  alter table G5\_COUNTRY ADD CONSTRAINT CONSTRAINT\_COUNTRY\_REGION CHECK ([COUNTRY\_REGION]='Oceania' OR [COUNTRY\_REGION]='Europe' OR [COUNTRY\_REGION]='Americas' OR [COUNTRY\_REGION]='Africa' OR [COUNTRY\_REGION]='Asia');  alter table G5\_COUNTRY ADD CONSTRAINT PK\_COUNTRY\_ID PRIMARY KEY clustered (COUNTRY\_ID);  -- 4. Create Products table  select \* into  G5\_PRODUCTS from  [dbo].[LI\_PRODUCTS]  -- 4.2 alter data definitions  go  sp\_rename 'G5\_PRODUCTS.PROD\_CAT\_DESC','PROD\_CATEGORY','COLUMN'  go  alter table G5\_PRODUCTS alter column PROD\_ID int not NULL;  alter table G5\_PRODUCTS alter column PROD\_NAME Varchar(50) not NULL;  alter table G5\_PRODUCTS alter column PROD\_SUBCATEGORY Varchar(50) not NULL;  alter table G5\_PRODUCTS alter column PROD\_CATEGORY Varchar(50) not NULL;  alter table G5\_PRODUCTS alter column PROD\_WEIGHT\_CLASS int not NULL;  alter table G5\_PRODUCTS alter column PROD\_LIST\_PRICE decimal(20, 2) not NULL;  alter table G5\_PRODUCTS ADD CONSTRAINT PK\_PROD\_ID PRIMARY KEY clustered (PROD\_ID);  -- 5. Create Promotions Table  --- 5.1 create and populate Promotions table  select \* into  G5\_PROMOTIONS from  [dbo].[LI\_PROMOTIONS]  --- 5.2 APPLY CONSTRAINT AND NEW DATA FILED DEFINITION  alter table G5\_PROMOTIONS alter column PROMO\_ID integer not null;  alter table G5\_PROMOTIONS alter column PROMO\_NAME varchar(30) not null;  alter table G5\_PROMOTIONS alter column PROMO\_SUBCATEGORY varchar(30) not null;  alter table G5\_PROMOTIONS alter column PROMO\_CATEGORY varchar(30) not null;  alter table G5\_PROMOTIONS alter column PROMO\_COST integer not null;  alter table G5\_PROMOTIONS alter column PROMO\_BEGIN\_DATE date null;  alter table G5\_PROMOTIONS alter column PROMO\_END\_DATE date not null;  alter table G5\_PROMOTIONS DROP column PROMO\_TOTAL;  alter table G5\_PROMOTIONS ADD CONSTRAINT PK\_PROMO\_ID PRIMARY KEY clustered (PROMO\_ID)  -- 6. Create Sales Table  --- 6.1 Create G5\_SALES table  select \* into  G5\_SALES from  (select \* from [dbo].[LI\_SALES\_12\_13]  union  select \* from [dbo].[LI\_SALES\_14]) a  order by [SALESTRANS\_ID]  go  --- 6.2 Table data definition  alter table G5\_SALES alter column SALESTRANS\_ID integer not null;  alter table G5\_SALES alter column PROD\_ID integer not null;  alter table G5\_SALES alter column CUST\_ID integer not null;  alter table G5\_SALES alter column CHANNEL\_ID integer not null;  alter table G5\_SALES alter column PROMO\_ID integer not null;  alter table G5\_SALES alter column SALE\_DATE date null;  alter table G5\_SALES alter column SHIPPING\_DATE date null;  alter table G5\_SALES alter column PAYMENT\_DATE date null;  alter table G5\_SALES alter column QUANTITY\_SOLD integer not null;  go  sp\_rename 'G5\_SALES.AMOUNT\_SOLD','TOTAL\_PRICE','COLUMN';  alter table G5\_SALES alter column TOTAL\_PRICE decimal(20,2) not null;  alter table G5\_SALES alter column UNIT\_PRICE decimal(20,2) not null;  alter table G5\_SALES ADD CONSTRAINT PK\_SALESTRANS\_ID PRIMARY KEY (SALESTRANS\_ID);  alter table G5\_SALES ADD CONSTRAINT FK\_PROD\_ID FOREIGN KEY (PROD\_ID) references G5\_PRODUCTS (PROD\_ID);  alter table G5\_SALES ADD CONSTRAINT FK\_CUST\_ID FOREIGN KEY (CUST\_ID) references G5\_CUSTOMERS (CUST\_ID);  alter table G5\_SALES ADD CONSTRAINT FK\_PROMO\_ID FOREIGN KEY (PROMO\_ID) references G5\_PROMOTIONS (PROMO\_ID);  alter table G5\_SALES ADD CONSTRAINT FK\_CHANNEL\_ID FOREIGN KEY (CHANNEL\_ID) references G5\_CHANNELS (CHANNEL\_ID) |

# Section 7: Database Implementation

## DML scripts to populate tables

All the scripts are shared on a github repository (https://github.com/axin-geo/IST302DBG5FINAL).

1. Populate G5\_CHANNELS

INSERT INTO [dbo].[G5\_CHANNELS]

SELECT [CHANNEL\_ID], [CHANNEL\_DESC], [CHANNEL\_DESC] FROM [dbo].[LI\_CHANNELS]

2. Populate G5\_COUNTRY

select DISTINCT [COUNTRY\_ID], COUNTRY\_NAME, COUNTRY\_SUBREGION, COUNTRY\_REGION

into G5\_COUNTRY

from G5\_CUSTOMERS

3. Populate G5\_CUSTOMERS

--- 2.1 Create G5\_CusEx table

select \* into G5\_CusEx

from [dbo].[LI\_CUSTOMERS\_EXT]

--- 2.2 Create G5\_CusIn table

select \* into G5\_CusIn

from [dbo].[LI\_CUSTOMERS\_INTX]

--- 2.3 Join tables together

select cx.[CUST\_ID], [CUST\_FIRST\_NAME],[CUST\_LAST\_NAME],

cx.[CUST\_GENDER],[CUST\_YEAR\_OF\_BIRTH],[CUST\_MARITAL\_STATUS],

[CUST\_INCOME\_LEVEL],[CUST\_CREDIT\_LIMIT],

[CUST\_MAIN\_PHONE\_NUMBER],[CUST\_EMAIL],

[CUST\_STREET\_ADDRESS],[CUST\_POSTAL\_CODE],

[CUST\_CITY],[CUST\_STATE\_PROVINCE],

[COUNTRY\_ID],[CUST\_TOTAL],

[COUNTRY\_NAME],[COUNTRY\_SUBREGION],

[COUNTRY\_REGION],[COUNTRY\_TOTAL]

into G5\_CUSTOMERS

from G5\_CusEx cx

join G5\_CusIn ci on cx.CUST\_ID = ci.CUST\_ID

4. Populate G5\_PRODUCTS

select \* into

G5\_PRODUCTS from

[dbo].[LI\_PRODUCTS]

5. Populate G5\_PROMOTIONS

select \* into

G5\_PROMOTIONS from

[dbo].[LI\_PROMOTIONS]

6. Populate G5\_SALES

select \* into

G5\_SALES from

(select \* from [dbo].[LI\_SALES\_12\_13]

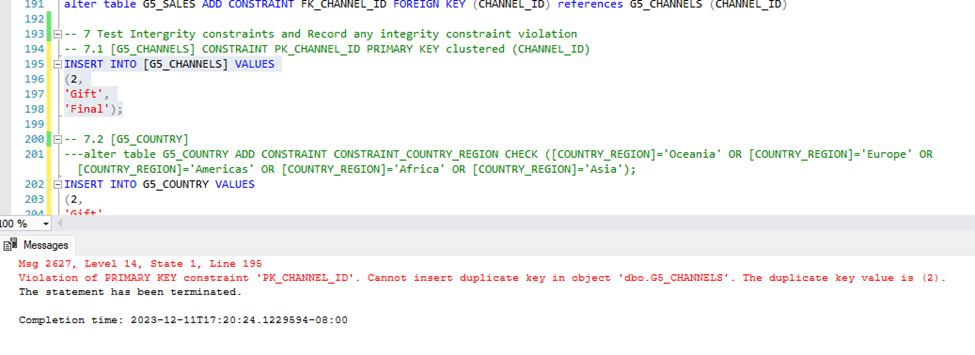
union

select \* from [dbo].[LI\_SALES\_14]) a

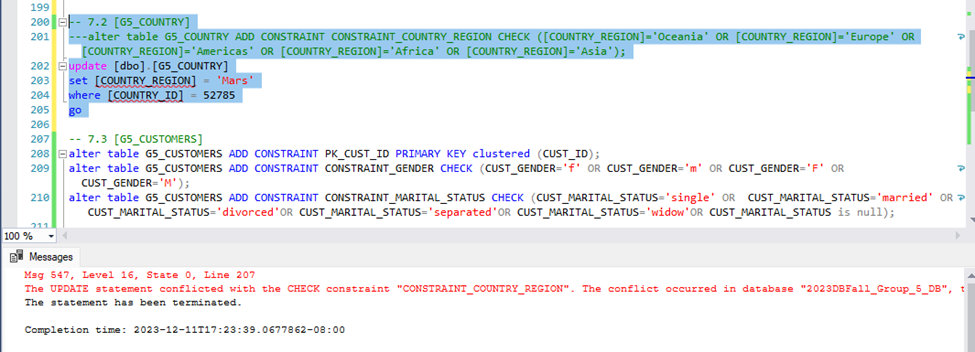
order by [SALESTRANS\_ID]

## Test integrity constraints

1. G5\_CHANNELS (A negative TEST ON PRIMARY KEY. A PRIMARY KEY HAS UNIQUE VALUES AND NOT NULL)

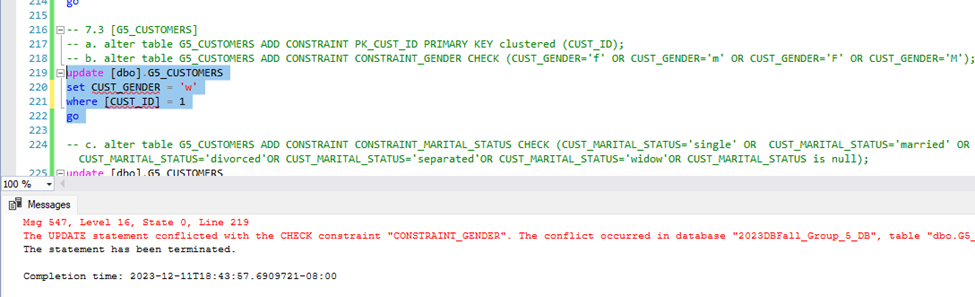


2. G5\_COUNTRY (This is a negative test on domain constraint. We added a constraint to 5 domains. If I try to update the COUNTRY\_REGION for one row, the domain constraint will be triggered and voided the operation.)

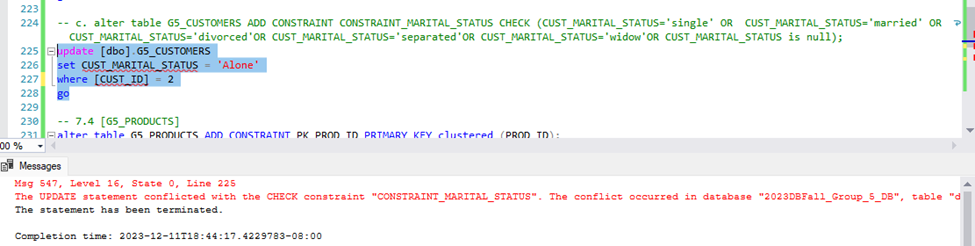


3. G5\_CUSTOMERS

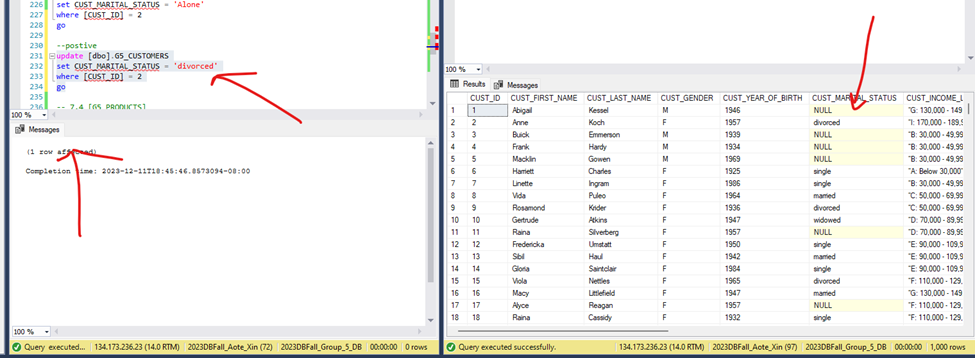
a. CONSTRAINT\_GENDER (Negative)



b. CONSTRAINT\_MARITAL\_STATUS (Negative)

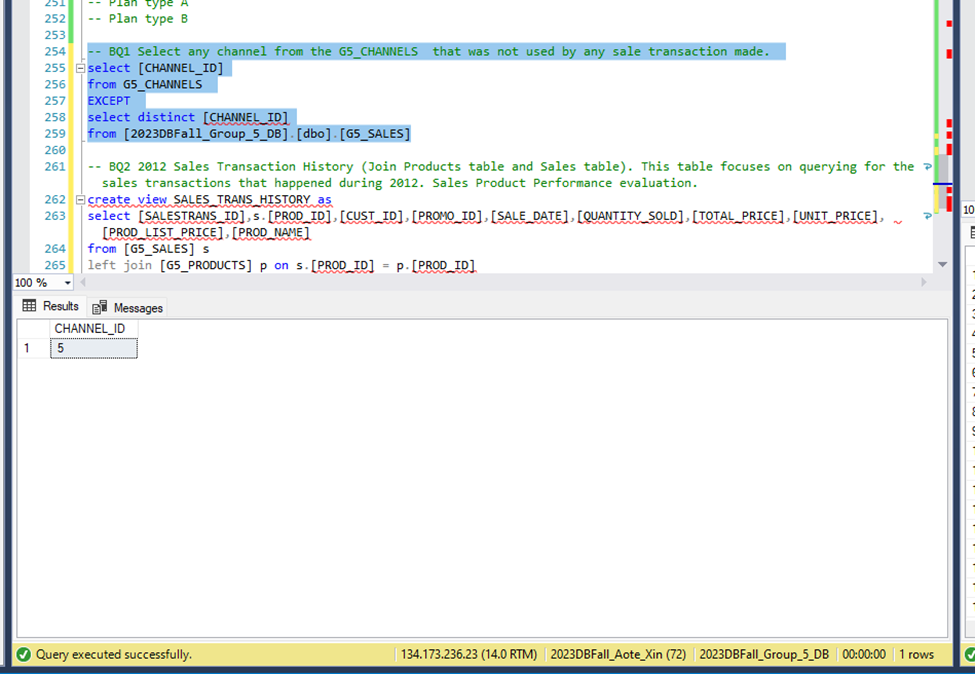


c. CONSTRAINT\_MARITAL\_STATUS (positive)

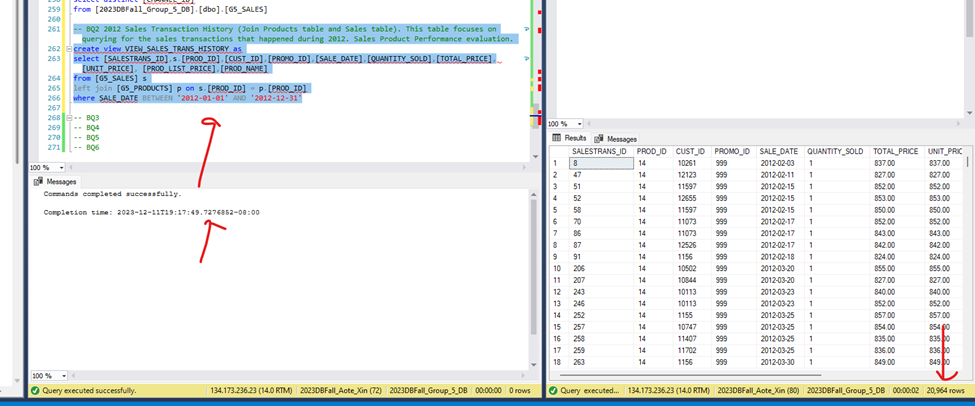


## Design and implement data access structure for the database system, including **,queries, views, indexes, stored procedures,** **security**

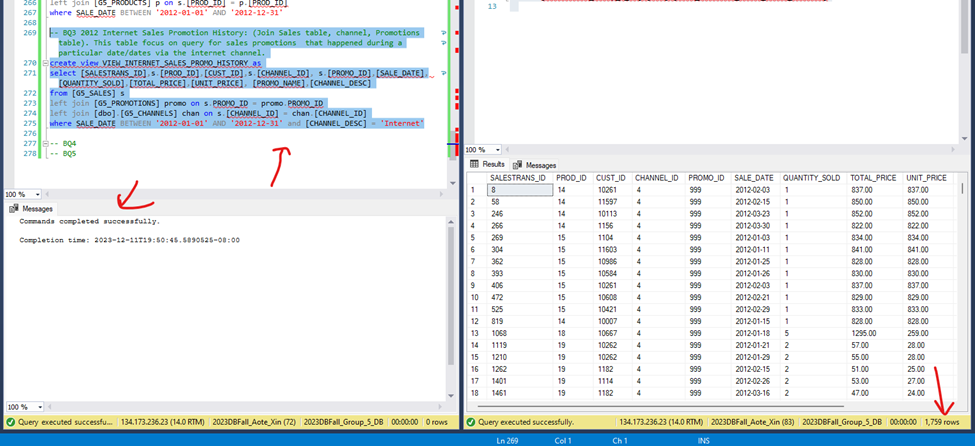
1. Business Query 1 (Select any channel from the G5\_CHANNELS that was NOT used by any sale transaction made.)



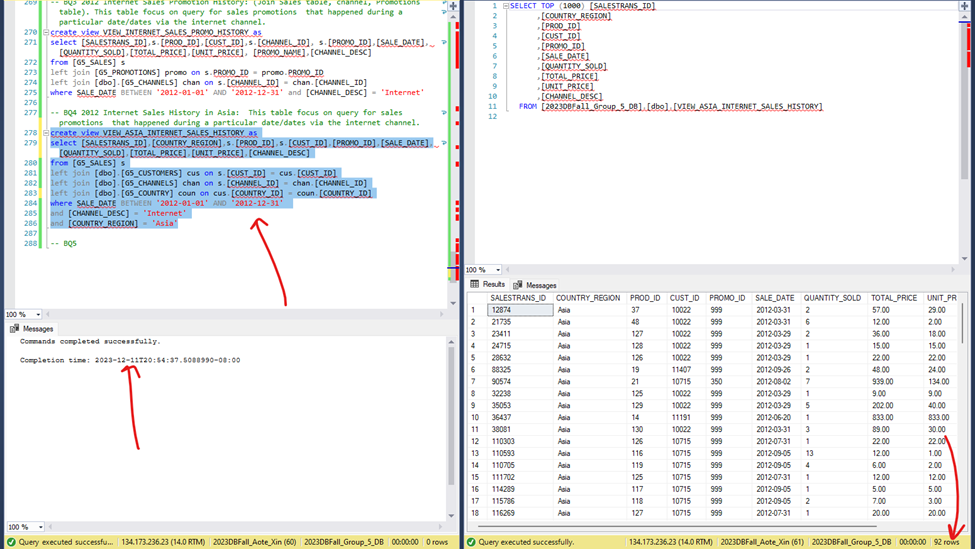
2. Business View 2 (2012 Sales Transaction History (Join Products table and Sales table). This table focuses on querying for the sales transactions that happened during 2012. Sales Product Performance evaluation.) Requirement: Two Tables - JOIN without or with condition, right or left join with complex condition.



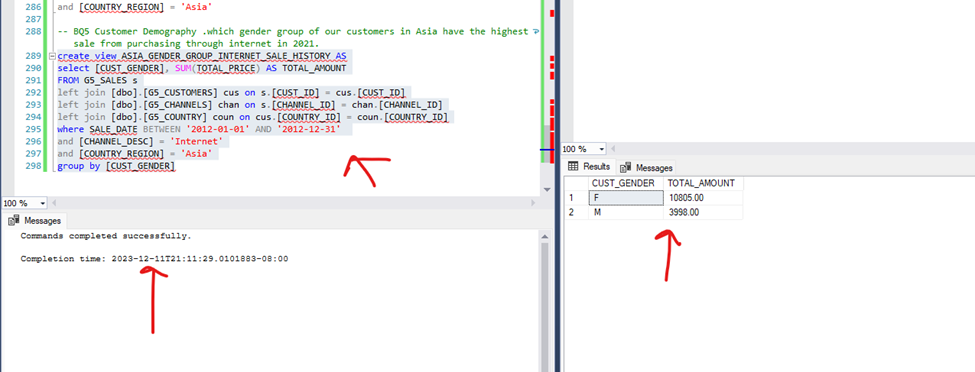
3. Business View 3 (2012 Internet Sales Promotion History: (Join Sales table, channel, Promotions table). This table focus on query for sales promotions that happened during a particular date/dates via the internet channel.) Required: Same as for Two Tables with at least one being either LI\_SALES\_12\_13 or LI\_SALES\_14.



4. Business View 4 (2012 Internet Sales Promotion History in Asia: This table focuses on query for sales promotions that happened during a 2012 via the internet channel in the continent of Asia) Required: 4 tables and need to use COUNTRIES Table.

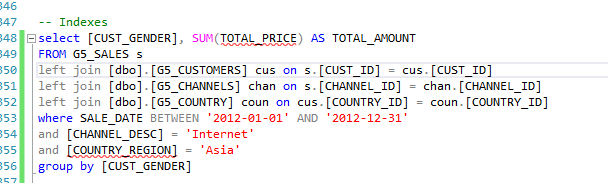


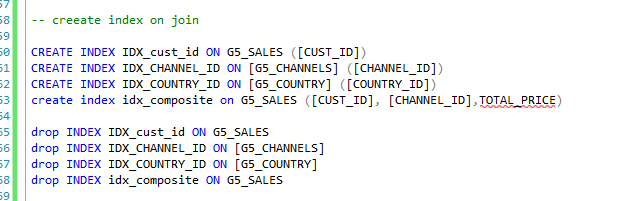
5. Business View 5 (Customer Demography .which gender group of our customers in Asia have the highest sale from purchasing through internet in 2021.) Required: Same as Detail Queries with Four Tables but involving SUM, AVG, COUNT, MAX, MIN operators.

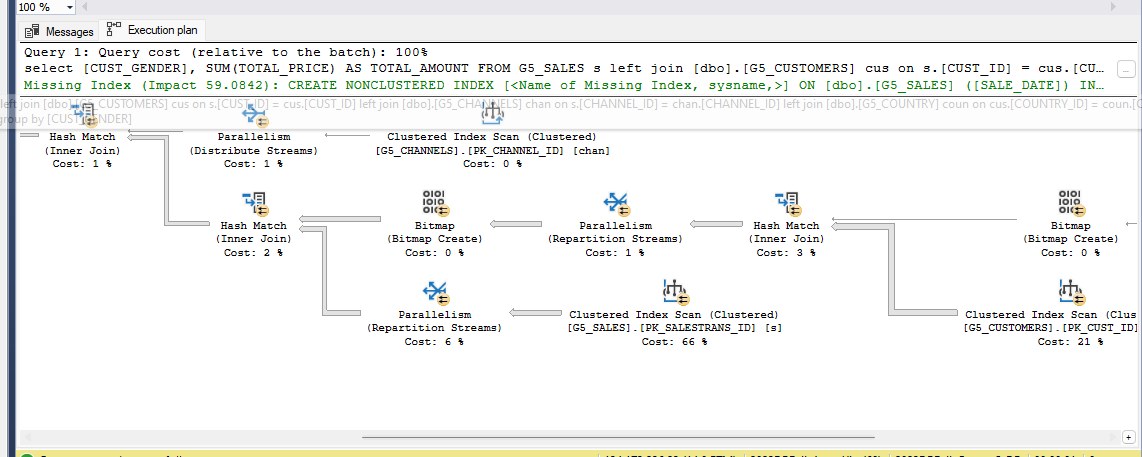


6. Indexes

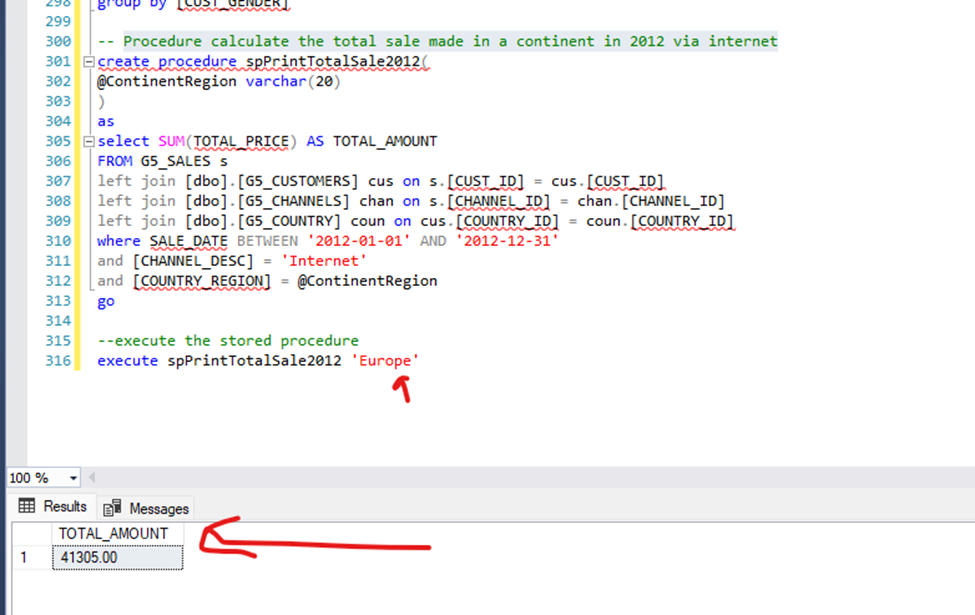
For the query below, we created 3 indexes and on composite index. Before creating the indexes the execution plan suggested that we creat a composite index on G5\_SALES for CUST\_ID, CHANNEL\_ID AND TOTAL\_PRICE. After we created the indexes as suggested, this new plan takes uses of hash join physical operator for inner join and take advantages of the non-clustered index for non-clustered index scan and index seek. Table scan was completed multiple time. And predicates were enforced during the table scan.



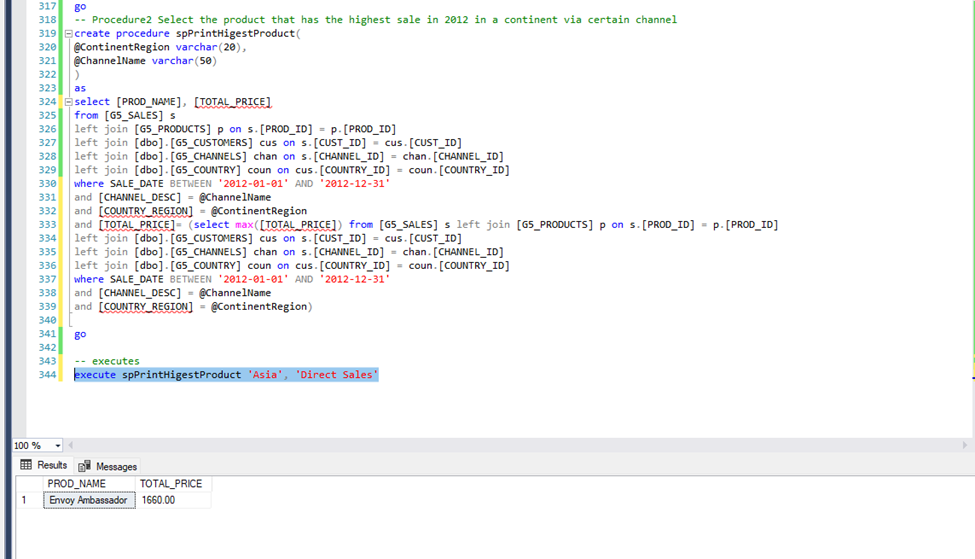




7. Stored Procedure 1 (Procedure1 calculate the total sale made in a continent in 2012 via internet)



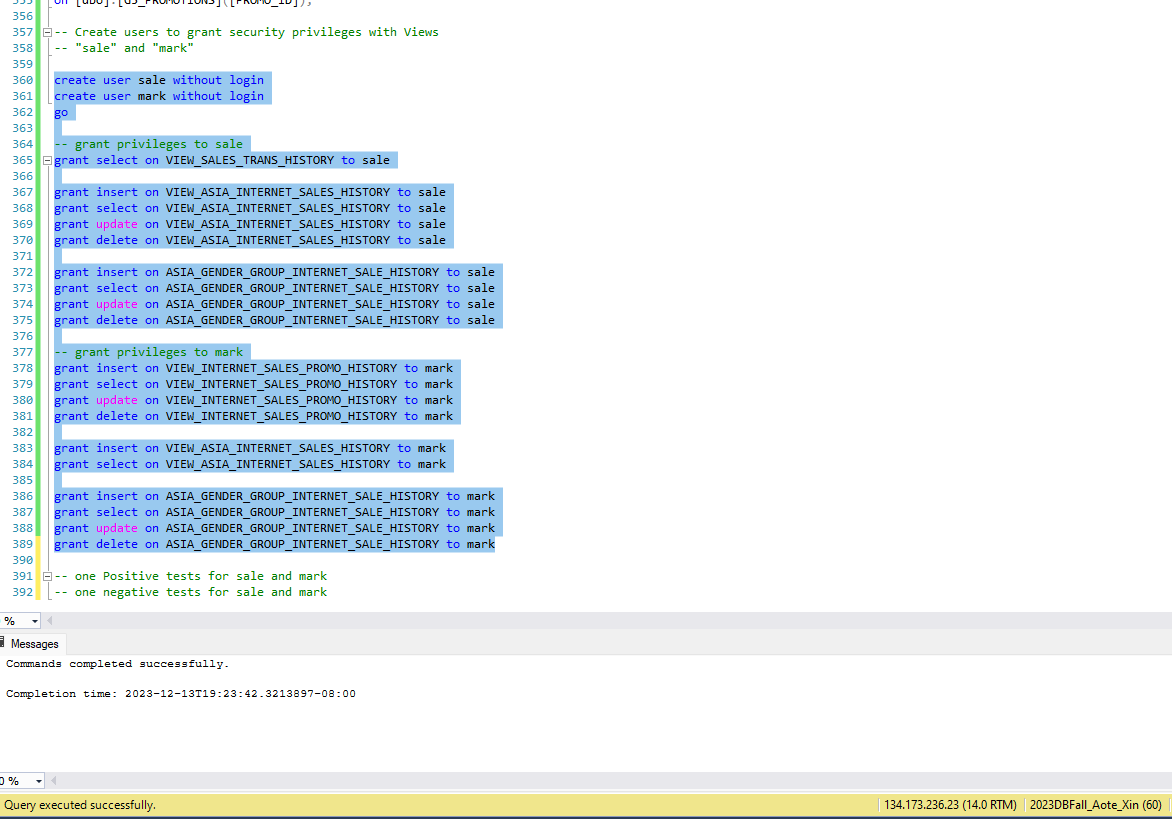
8. Stored Procedure 2 Select the product that has the highest sale in 2012 in a continent via certain channel



9. Create users (Sale and Mark) to grant security privileges with Views

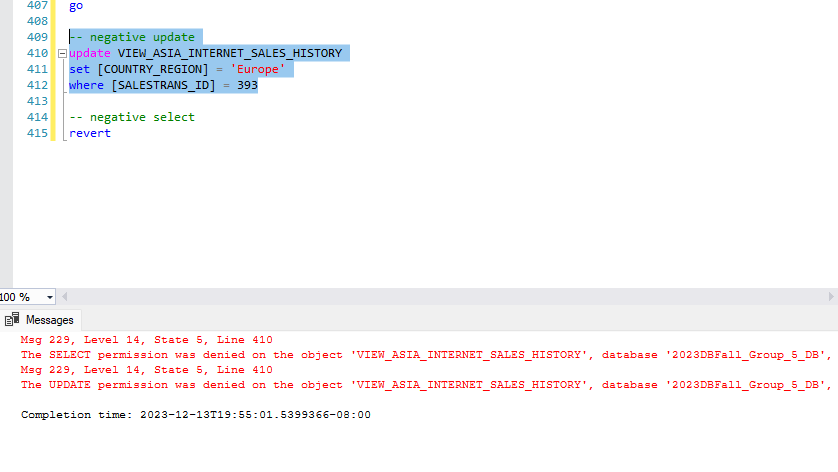
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***BUSINESS VIEWs*** | | | | |
| ***USERs*** | BV02 (VIEW\_SALES\_TRANS\_HISTORY) | BV03 (VIEW\_INTERNET\_SALES\_PROMO\_HISTORY) | BV04 (VIEW\_ASIA\_INTERNET\_SALES\_HISTORY) | BV05 (ASIA\_GENDER\_GROUP\_INTERNET\_SALE\_HISTORY) |
| Sales (sale) | S |  | ISUD | ISUD |
| Marketing (mark) |  | ISUD | IS | ISUD |

9.1 Create the users and grant security privileges to sale and mark according to deign

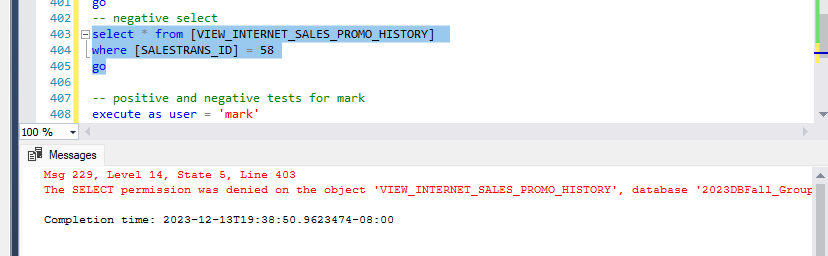


9.2 positive and negative tests for sale

9.2.1 negative update

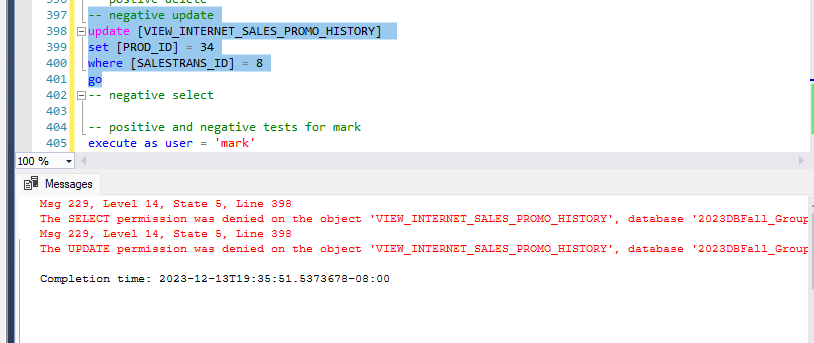


9.2.2 negative select

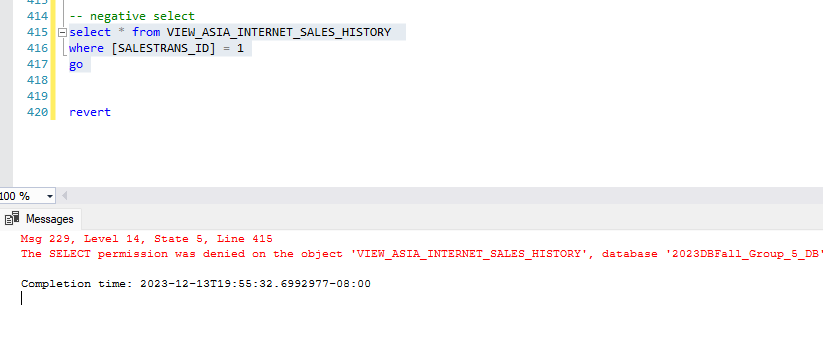


9.3 positive and negative tests for mark

9.3.1 negative update



9.3.2 negative select



# Section 8: Conclusion

Through this database project, we've deepened our understanding of database design, including selecting the appropriate data types for each attribute and the importance of data cleansing to maintain a robust database. We've gained practical experience in constructing business views, setting up correct access controls, and specifying user permissions, ensuring that only relevant users are included in the database system. By developing integrity constraints, we were able to align our technical knowledge with specific business requirements, enforcing limits where necessary to safeguard the data integrity. The hands-on aspects of the project, including creating and consolidating tables, loading data, implementing stored procedures, testing constraints, and executing queries, allowed us to apply our theoretical planning to a practical environment. We learned not only to create ER diagrams and discern redundant data but also to consolidate tables with overlapping functions, thereby optimizing database size and utility. By employing more suitable data types, we could save space and reduce errors. Establishing meaningful relationships between tables was instrumental in making the database functional and in ensuring it provides relevant and actionable insights.

References:

* <https://www.ibm.com/docs/en/cics-ts/5.4?topic=samples-sale-example-application>
* <https://smallbusiness.chron.com/account-cash-receipts-76317.html>