In partial fulfillment of the requirement for
System Development
420-940-VA section 05808
Vanier College

Database Design

6th Deliverable

Customer Name:
Delhi Kabab

Team Name:
Green Team

Team Leader:
Bermet Doolbekova

Team members:

I, Lei Wang, ID: 2295046, certify that I contributed to this deliverable

I, Bing Bai, student ID# 2295011, certify that I contributed to this deliverable

I, Shayekh Tarafdar, student ID# 2295035, certify that I contributed to this deliverable

I, Md Ibrahim Ullah, student ID# 2295025, certify that I contributed to this deliverable

I, Bermet Doolbekova, student ID# 2295017, certify that I contributed to this deliverable

I, Mark Benedict Muyot, student ID# 2295022, certify that I contributed to this deliverable

Date of Submission:
November 11, 2022

TABLE OF CONTENTS

1. STATEMENT OF PREVIOUS WORK EXPERIENCE

In this deliverable, the team worked on the database design for the inventory management application for Delhi Kabab. The team was able to what they learned from database subject. Below is a list of project/s each member worked on in relation to designing database.

Table 1: Prior projects with related to UI designs

Projects Made	**Skills**	**Contributor/s**
Cars2Go	Relational Database Model, ER Diagram, Relationship Schema, Microsoft SQL Server, Entity Relationship Diagram, Table Normalization, SQL Query, SQL Script	Everyone

2. EXECUTIVE OVERVIEW

In this deliverable, the team worked on the database design for Delhi Kabab's inventory management application. The team used relational database model for the database structure and draw.io in creating the diagrams needed. The team stated the updated business problem of their client and a narrative description of the database design in this deliverable.

The group was able to create a block diagram that illustrates how users interact with various components of the database using the application. The team was also able to provide data dictionary showing all entities, relations and an ER-diagram of the inventory management system complete with cardinalities **(see Appendix A & B).**

The database architecture, indexes, and hot to optimize database query were also discussed in this deliverable **(see Appendix C)**. In addition, the team was also made a projection on how big the database will be in three-to-five-year time. A detailed calculation and assumptions will be seen in the appendix section **(see Appendix D)**.

Finally, the response time of the database was also determined and how often will it be used was discussed also in the appendix section **(see Appendix E).**

3. BUSINESS PROBLEM

Delhi Kabab has an inefficient system in monitoring their inventories. The restaurant is using a notepad to list necessary items to be ordered such as ingredients for their food preparations, cleaning items, toilet supplies etc. According to the client, they often lose their list and sometimes forget to update the list of ordered items in their system. In consequence, the restaurant's inventory record often produces inconsistent information. This type of system leads to increase in spoilage and loss if inventories like ingredients are not tracked and monitored properly. The system also takes a lot of time tracking and reviewing their inventory. There is no way to view a comprehensive report easily, such as the ingredient supply level.

Customer satisfaction is sometimes affected by the inconsistencies in their inventory management system. There are times where customers want to order something, but the food is not available because some ingredients are missing. This leaves unsatisfied customers which could affect the bottom line of the business.

Lastly when it comes to security, their excel file for tracking inventories is not encrypted or password protected. This can be easily access and modify by anyone within the organization. The owner realized that this type of system can lead to pilferage without him knowing it. Someone can just manipulate the stock level and steal some inventories used in the restaurant.

4. NARRATIVE DESCRIPTION OF DATABASE DESIGN

The team implemented a relational database model for the database design since it is scalable and extensible which makes it easy to provide a flexible structure to meet changing requirements and increasing amounts of data.

In the inventory management system of the restaurant, the admin and the guest are the actors that will interact with the system. Each actor has different access level. The admin is given a full access privilege where he can perform certain CRUD operations in managing the account in the system. For example, the admin can create, add, delete, or modify user accounts in the database. The admin is also capable of performing CRUD operations in monitoring and tracking items in their inventories like adding, deleting, updating an item in the database. The admin is the only user that could also create and modify order list and supplier's database.

Meanwhile, the guest can only view, read, and search items in the inventory. The guest is not allowed to do modifications that could change the value of the data in the database.

5. BLOCK DIAGRAM

6. APPENDIX A: DATA DICTIONARY, RELATIONSHIP/MULTIPLICITY & ATTRIBUTES TABLE

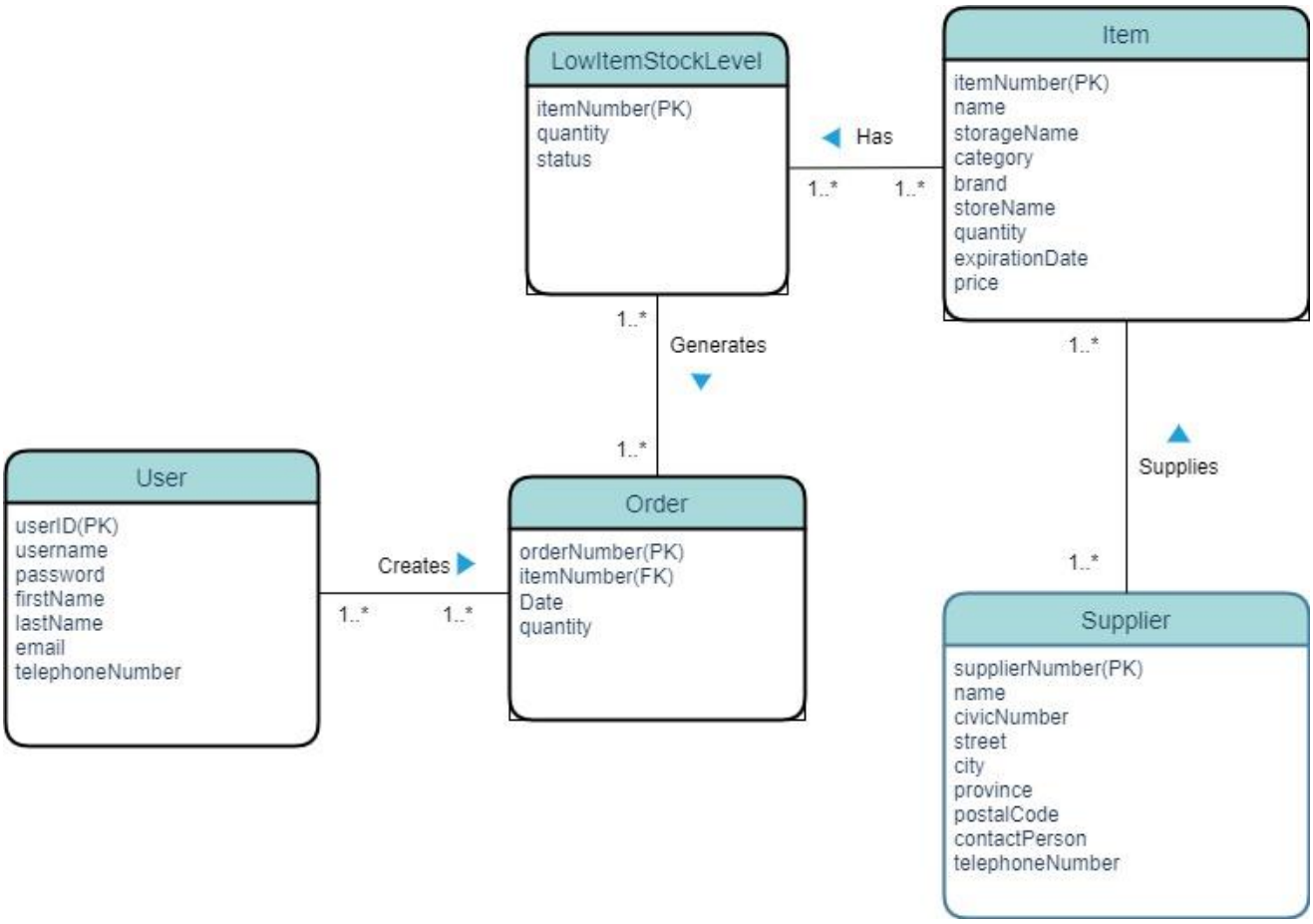
Data Dictionary		
Entity	**Description**	**Aliases**
User	Individuals who will be accessing the application	na
Order	List the needs to be bought or order	na
LowItemStockLevel	Items that are currently running low in stock	na
Item	Items that being stored and tracked by Delhi Kabab	na
Supplier	Delhi Kabab's supplier	na

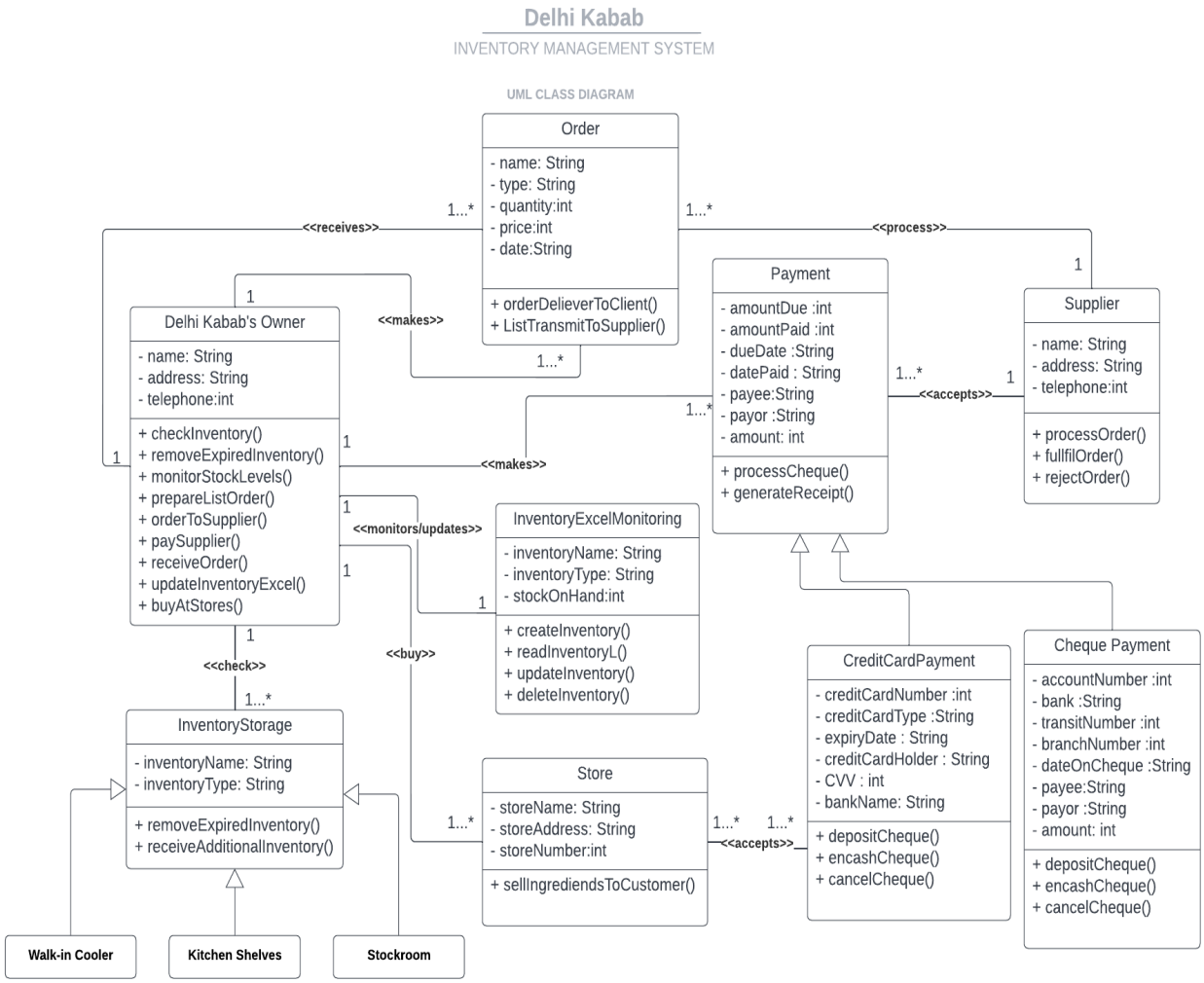
Relationship/Multiplicity				
Entity	**Multiplicity**	**Relationship**	**Multiplicity**	**Entity**
User	1..*	Creates	1..*	Order
LowItemsStockLevel	1..*	Generates	1..*	Order
Item	1..*	Has	1..*	LowItemsStockLevel
Supplier	1..1	Supplies	1..*	Item

Attributes Table								
Entity	**Attributes**	**Primary key**	**Foreign Key**	**DESCRIPTION**	**DATA TYPE & LENGTH**	**Null**	**SINGLE-VALUED**	**MULTI-VALUED**
User	userID	Yes	No	uniquely identifies the user	INTEGER()	No	Yes	No
	username	No	No	username used to login to the application	VARCHAR(20)	No	Yes	No
	password	No	No	password to login to the application	VARCHAR(20)	No	Yes	No
	firstName	No	No	First name of the user	VARCHAR(15)	No	Yes	No
	lastName	No	No	Last name of the user	VARCHAR(15)	No	Yes	No
	dateOfBirth	No	No	Date of birth of the user	DATE	No	Yes	No
	email	No	No	Email address of the user	VARCHAR(30)	No	Yes	No
	telephoneNumber	No	No	Telephone number of user, it could be mobile or landline.	INTEGER()	No	No	Yes
order	OrderNumber	Yes	No	Identifies the order	INTEGER()	No	Yes	No
	ItemNumber	No	Yes	Identified the item in the order	INTEGER()	No	Yes	No
	Date	No	no	Date of the order placed	DATE	No	Yes	No
	Quantity	No	no	Quantity of the items in the order	INTEGER()	No	Yes	No
LowitemStockLevel	ItemNumber	Yes	No	Identifies the unique item	INTEGER()	No	Yes	No
	Quantity	No	No	Quantity of the items in stock	INTEGER()	No	Yes	No
	Status	No	no	Status of the order (pending, submitted, completed)	VARCHAR(15)	No	Yes	No

item	itemNumber	Yes	No	Identifies unique item	INTEGER()	No	Yes	No
	name	No	No	Name of the item	VARCHAR(30)	No	Yes	No
	storageName	No	No	Storage where the item is placed	VARCHAR(30)	yes	No	No
	category	No	No	Which category the item belongs to	VARCHAR(30)	No	Yes	No
	brand	No	No	Brand of the item	VARCHAR(30)	No	Yes	No
	storeName	No	No	Store where the item was bought	VARCHAR(30)	yes	yes	no
	quantity	No	No	Quantity of the items remaining in the stock	INTEGER()	No	Yes	No
	expirationDate	No	No	Date when the item will expire	DATE	No	Yes	No
	Price	No	No	price of the item	MONEY	No	Yes	No
supplier	supplierNumber	Yes	No	Identifies the supplier	INTEGER()	No	Yes	No
	name	No	No	Name of the Supplier	VARCHAR(30)	No	Yes	No
	civicNumber	No	No	Civic number of the supplier's address	INTEGER()	No	Yes	No
	street	No	No	Street name of the address	VARCHAR(30)	No	Yes	No
	city	No	No	City where the supplier operates from	VARCHAR(30)	No	Yes	No
	province	No	No	Province where the supplier operates from	VARCHAR(30)	No	Yes	No
	postalCode	No	No	Postal code of the supplier's address	VARCHAR(7)	No	Yes	No
	contactPerson	No	No	Contact person when user contacts the suppliers	VARCHAR()	No	Yes	No
	telephoneNumber	No	No	Phone number of the company	INTEGER()	No	No	Yes

7. APPENDIX B: ER-DIAGRAM





Comparison between ER diagram and Class diagram

In class diagram multiple classes were shown such as order, owner, supplier, payment which includes credit card, cheque payment, etc. On the other hand, the ER-Diagram represented fewer entities such as supplier, order, user, and item. Most components in the UML class were not presented in the ER-Diagram because the team tried to capture the whole operating picture of Delhi Kabab from inventory management system to payment system which is already part of the accounting system.

Another reason why both diagrams looked different, class diagram are more likely to show real world object with its attributes and behaviors while ER-diagram often maps tables in the database and entities interacting within the database. ER- diagram cannot capture the functions or behaviors of each entity unlike in class diagram.

The two diagrams have commonalities even though they appear to be different. The user entity in the ER-diagram represents the owner in the class diagram. The attributes in the ER-diagram were broken down into atomic attributes unlike in the class diagram but both pertains to the name of the users including the owner. For example, in class diagram there is a name attribute while in ER-diagram it was broken down to first name and last name. This was done to facilitate querying of data. Multiple valued data type should be normalized into atomic or single valued data.

Common elements such as order, supplier, and item (which represents inventory in the class diagram) are both presented in the diagrams. Some important attributes that were not included in the class diagram were added in the ER-Diagram. These attributes will be used in the system. For example, supplier in the class diagram don’t contain contact person while in ER-diagram, the supplier entity contains contact person attribute.

8. APPENDIX C: INDEXES, DATABASE ARCHITECTURE & QUERY OPTIMIZATION

Indexes and Architecture

Primary Key Indexing was used in this project. This indexing method is the default method of Microsoft SQL Server. This indexing method is automatically created for the primary key to ensure that a record is unique. This will ensure the data integrity of the database. This indexing technique assures any query to be very efficient (Nath, 2019). This will also improve performance when looking for records, sorting records, grouping records, or keeping a separate column.

For the Architecture, relational database model was implemented. Sorting data is made simpler by the relational approach. This aids in enhancing database querying.

Query Optimization

Some query optimization techniques will be used to speed up the search process like using inner join instead of outer join since this limits database optimization which may result to slower SQL execution. Avoiding unnecessary columns in SELECT clause and targeting needed attributes in a table instead of targeting all attributes will also help in boosting up the speed like. The sample below shows a query on Delhi Kabab's Supplier's table not selecting all attributes by being selective on the needed attributes only.

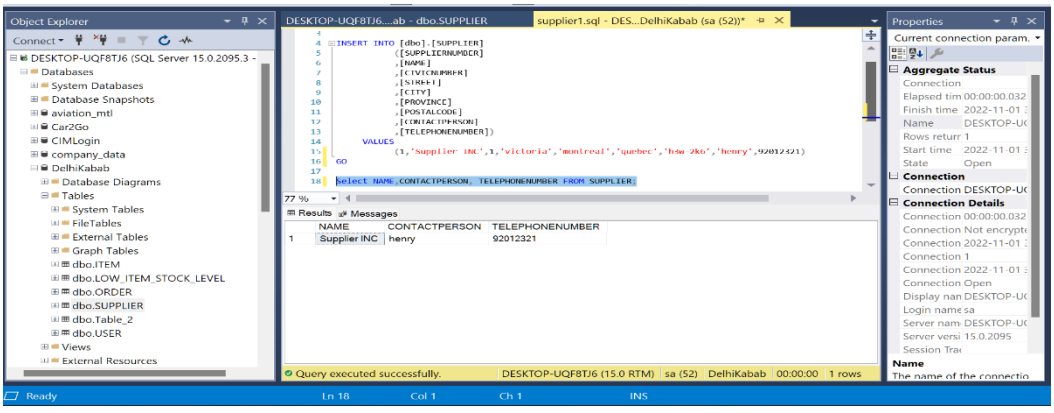


Figure 1 Query Supplier's Table with Selected Attributes

9. APPENDIX D: PROJECTED SIZE OF THE DATABASE

To be able to make a projection, the team made a simulation and created the database with 5 tables according to their ER-Diagram. There were 5 entities in the ER-Diagram that were converted into tables. The team used Microsoft SQL Server in their simulation and established the following assumptions to determine how many records each table should have.

Table Name	Number of Attributes	Size per record (Computed based on attribute's table)	Assumption	Projected Number of Records in 5 years	Table Size based on a 5-year projection
Supplier	9	111 Bytes	Delhi Kabab has one supplier. The team made a projection to add 4 more records in the supplier's tble within span of 5 years	5	111 Bytes * 5 Records =555 Bytes
User	7	15 Bytes	Admin is only one but incase another admin account is needed then admin can create another admin account. Assigning to many admin is not necessary. Maximum of 4 admin could be assigned within 5 years then	4	15 Bytes * 4 Records =444 Bytes
Item	7	23 Bytes	According to the client they got approximately 100 -150 type of items they are storing in their restaurant. The team simulated and added 150 records in the item's table **(see Appendix F)**.	150	23 Bytes * 150 Records =3,450 Bytes
LowItemStockLevel	3	149 Bytes	In this table, the team simulated 30 records in case the restaurant gets 30 items with low stock level per month. Since this is computed per month then it should be multiplied to 60 to get the 5-year projection (**see Appendix F)**. **30 * 60months=1800 records**	1800	148 Bytes * 1800 records =266,400 Bytes

Order	4	159 Bytes	The team projected 40 types of items being ordered per month. 2400 random records were generated to cover the 5-year projection. (**see Appendix F**). **40*60=2400 records**	2400	159 Bytes * 2400 records =381,600 Bytes
Total memory needed (5-year projection):					**652,449 Bytes or 652.45 KB**

In term of the table size based on a 5-year projection, 652.45 KB of memory should be allocated. The team also did a simulation by adding 4,359 random records (based on the total records for a 5-year projection) in the database. Approximately 80 MB should be allocated for the database. There might be components included when using SQL server management that is why it consumed 80 MB of memory. A screen shot of the database size was taken after adding all dummy records.

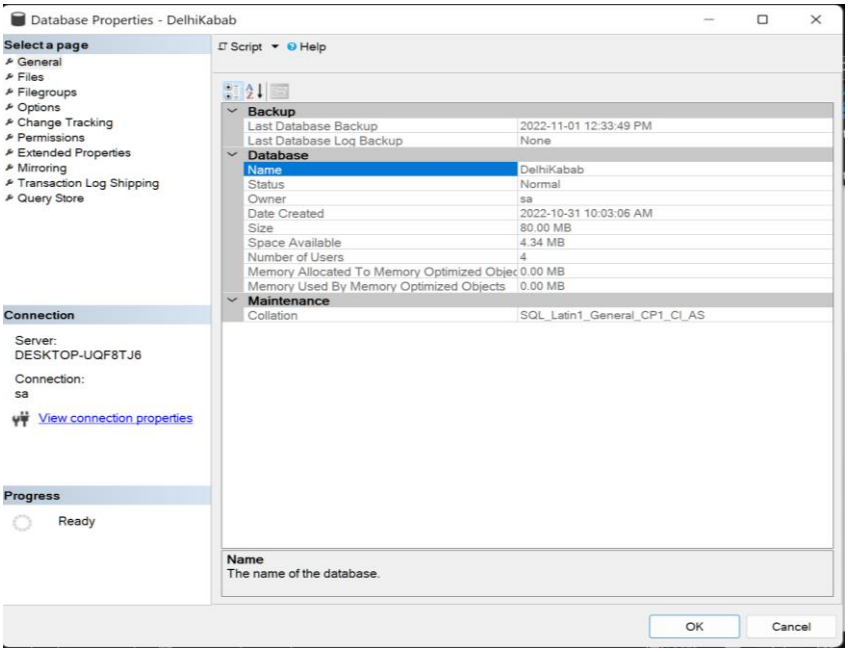


Figure 2 Database Size After Adding 4,359 Records(80MB)

10. APPENDIX E: DATABASE RESPONSE TIME

To ensure the speed and response time of the database, the team did a database simulation. The client also will just use the application three times a day to once a day and the application will be access through a solid-state drive which will guarantee a good speed when the client tries to access the application.

For the simulation, the team tested the table with most records based on a 5-year projection. 2,400 random records were added to order's table and performed a query to test the response time. The team performed a SELECT query, selecting all 24,000 records. The response time was 0.119s.

According to the client, a 2-3 second waiting time is manageable. With the results of the simulation, the team could guarantee that the response time would be faster than the 2-3 second threshold of the client. In addition, the client is using an Intel i7 processor with 8GB RAM. This could assure that response time will be faster than 0.119s because the simulation was done on an i5 processor with 4GB RAM only.

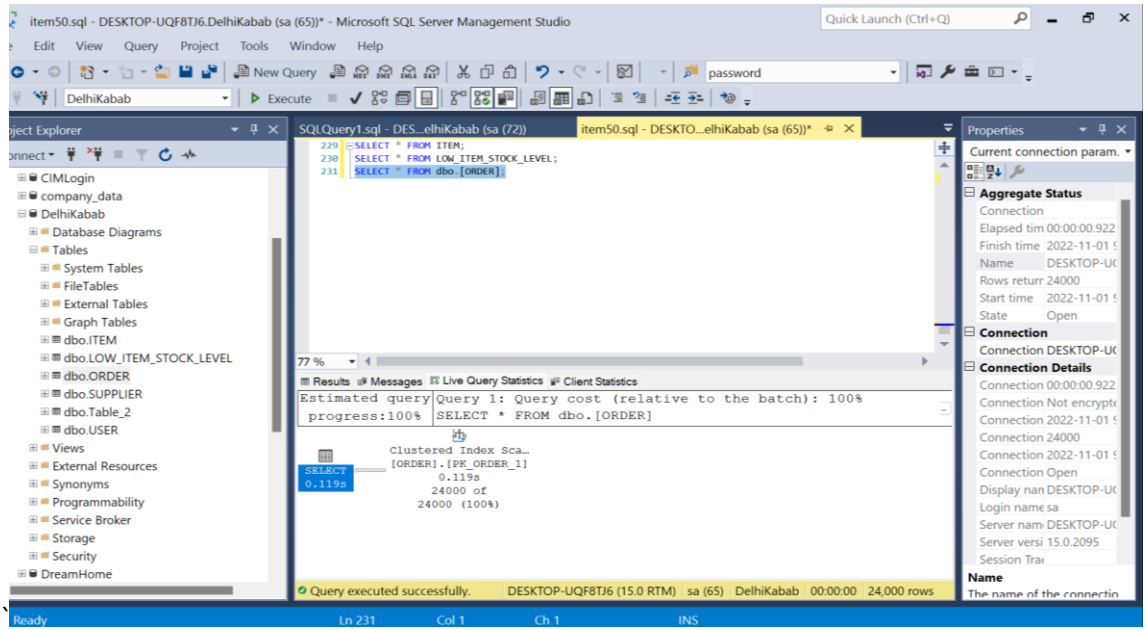


Figure 3 Select All 2,400 Records in Order Table (Response Time 0.119s)

11. APPENDIX F: TABLE SIMULATION

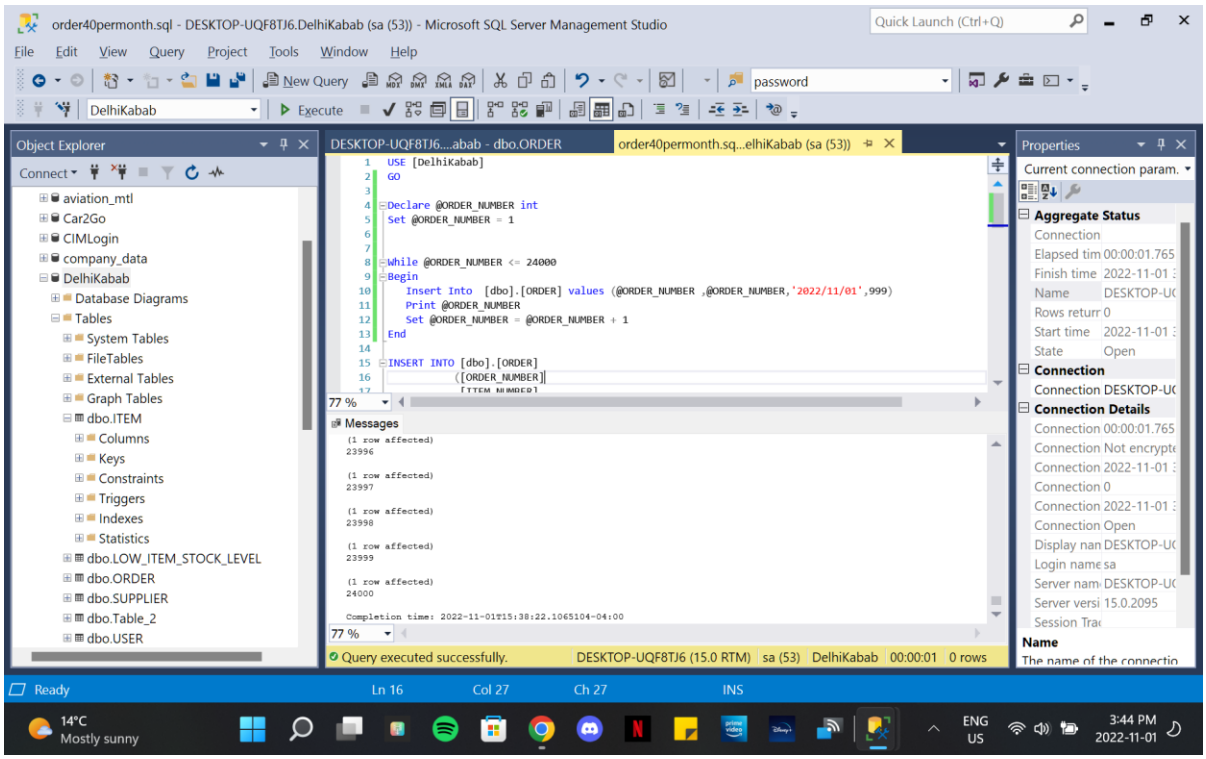


Figure 5 Added 2400 Random records in Order's Table

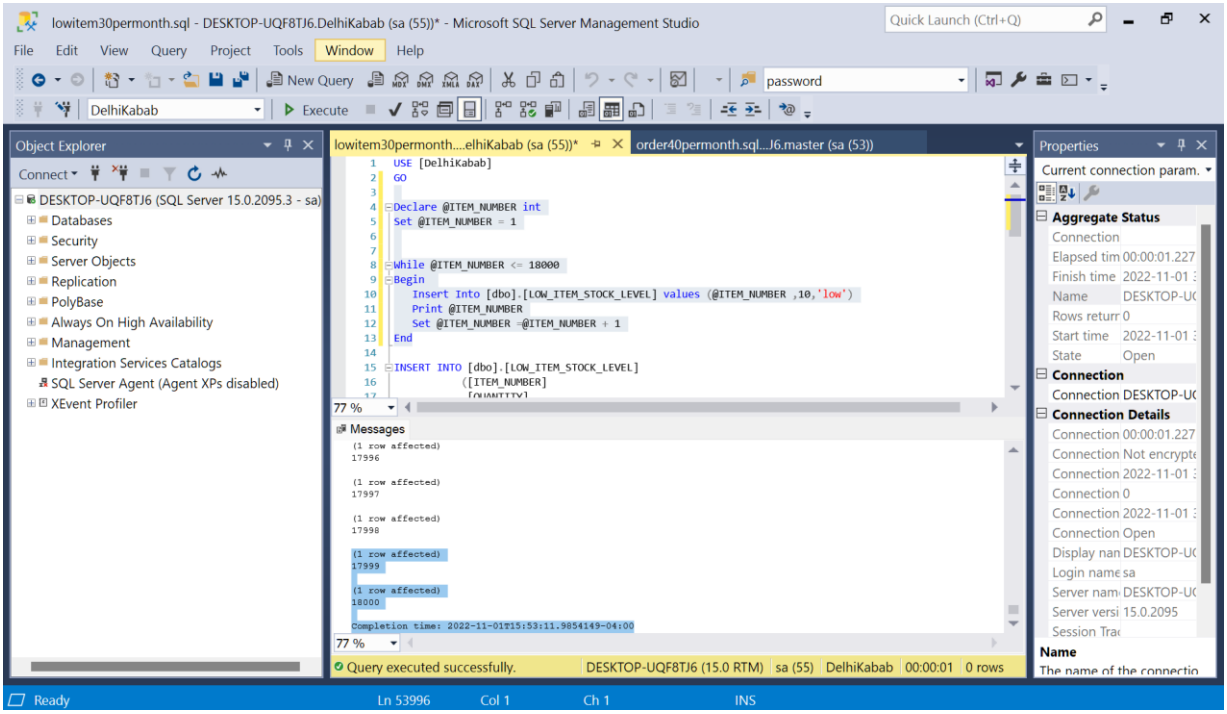


Figure 4 Added 1800 random records in LOWITEM TABLE

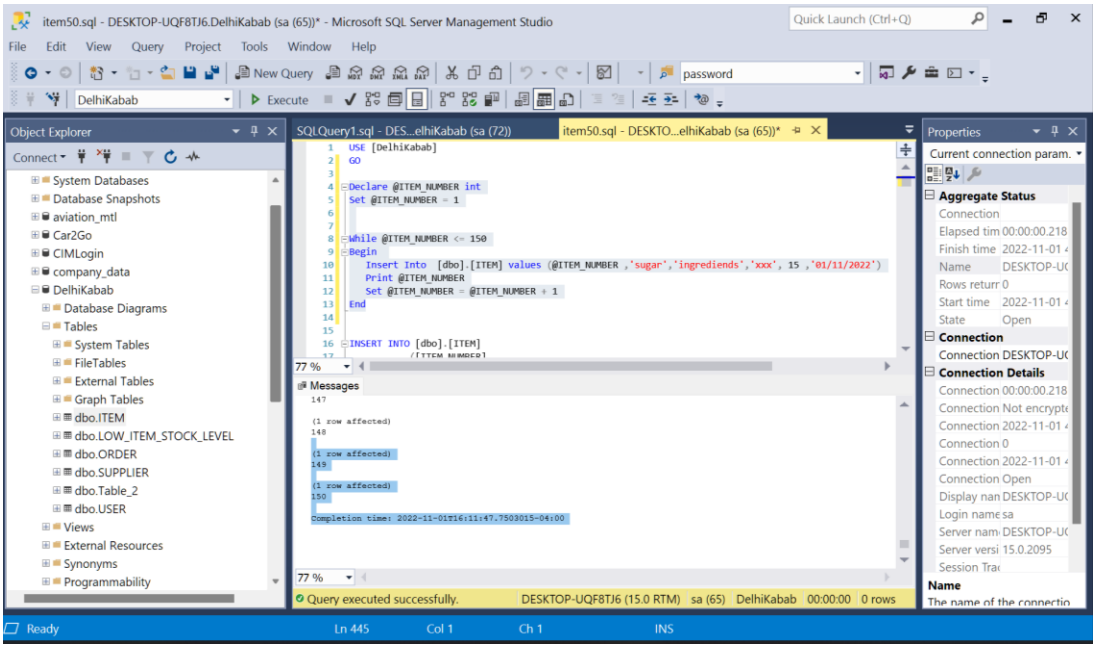


Figure 6 Added 150 items in ITEM's Table

12. REFERENCES

Nailimov, C. (June 09, 2020). *"Composite and other attributes in the ERD".* Gleek.

https://www.gleek.io/blog/er-model-attributes.html

Teorey, T. (2011). *"Multivalued Attributes".* ScienceDirect.

https://www.sciencedirect.com/topics/computer-science/multivalued-attribute

Glen, S. (July 25, 2021). *"E-R Diagram Cardinality and Participation".* Easy Agile.

https://www.datasciencecentral.com/e-r-diagram-cardinality/

Nishadha, R. (October 2022). *"Ultimate Entity Relationship Diagram Tutorial".* Creately.

https://creately.com/blog/diagrams/er-diagrams-tutorial/

Nath, K. (April 2019). "*An in-depth look at database Indexing*". FreeCodeCamp.

https://www.freecodecamp.org/news/database-indexing-at-a-glance-bb50809d48bd/#:~:text=Advantages%20of%20Primary%20Index%3A,clustered%20%26%20records%20are%20ordered%20physically.