Written by: Omphemetse Lekaba

DATABASES MODULE WEEK 1

Part 1: Understanding SQL

Answers:

Question 1. Research

1.1. Over 10 online stores come to mind: [1]

1. Amazon
2. eBay
3. Alibaba
4. Walmart
5. Apple Store
6. Zara
7. ASOS
8. Best Buy
9. Etsy
10. Nike

SQL enables these organizations to organize data into tables, ensuring that every product is categorized, described, and priced accurately. SQL offers a framework for designing, altering, and managing database structures. Using SQL, one can define tables, establish connections between them, and enforce rules to maintain data consistency.

1. Product Information: Stored in a products table, including details like name, description, price, and inventory.
2. User Accounts: Managed in a user's table with information such as usernames, emails, hashed passwords, and shipping addresses.
3. Order Details: Recorded in an orders table, encompassing data like order IDs, user IDs, product IDs, quantities, total amounts, and shipping information.

[1] L., D. (2024, March 25). SQL: The backbone of data-driven decision making. Retrieved June 22, 2024, from https://www.linkedin.com/pulse/sql-backbone-data-driven-decision-making-daniel-lozovsky-mba-fs8ec/

1.2. SQL[2] plays a crucial role in web applications by providing a robust mechanism for storing, retrieving, and manipulating data. It allows web developers to create and manage databases where they can store user information, product catalogues, session data, and more. Through SQL queries, web applications can dynamically fetch data based on user inputs, generate personalised content, and facilitate transactions securely. This structured approach to data management ensures that web applications can handle large volumes of data efficiently while maintaining data integrity and supporting rapid development cycles.

1.3. Benefits of using SQL for web applications.[3]

1. Commonality: SQL is widely used across IT systems and integrates well with various programming languages, ensuring its usefulness throughout professionals' careers and improving business efficiency.
2. Simplicity: SQL commands use straightforward English phrases, making them easy to understand for programmers and enabling quick learning for newcomers with limited coding experience.
3. Integration: SQL integrates smoothly with languages like Python and R, allowing seamless data manipulation and efficient database management in unified coding environments, particularly useful for data analysts, engineers, and web developers

1.4. Efficiency, Data Organization, And Data Retrieval Capabilities.[4]

1. Efficiency: SQL helps handle data swiftly, making operations on large datasets faster and improving overall system performance.
2. Data Organization: SQL allows for organizing data in a structured manner by defining tables, relationships, and rules within databases. This ensures that data is stored in a logical way that meets business requirements.

[2] McKay, Sam. "What Is SQL Used For? 7 Top Uses." *SQL*, Enterprise DNA, retrieved https://blog.enterprisedna.co/what-is-sql-used-for/.

[3] Indeed Editorial Team. (March 11, 2023). 10 Uses of SQL (With Definition, Benefits and Examples). Retrieved from https://www.indeed.com/career-advice/career-development/sql-uses

[4] L., D. (2024, March 25). SQL: The backbone of data-driven decision making. Retrieved June 22, 2024, from https://www.linkedin.com/pulse/sql-backbone-data-driven-decision-making-daniel-lozovsky-mba-fs8ec/

3. Data Retrieval Capabilities: SQL's querying abilities enable precise data retrieval by filtering, sorting, and aggregating data from multiple tables. This makes it easier to analyze data and generate insights that drive decision-making.

1.5. Database Management Systems (DBMS):

1. NoSQL Databases (e.g., MongoDB, Cassandra)
2. Excel and Spreadsheets
3. Business Intelligence (BI) Tools (e.g., Tableau, Power BI)

Part 2: Database Fundamentals

Question 2.1: Tables

A database table organizes information similarly to a spreadsheet, using rows and columns. Each row holds a single piece of data, while each column defines a specific category or detail. Tables, like spreadsheets, help manage data by making it easy to store, find, and change information. However, database tables are more powerful for handling large amounts of data and ensuring data accuracy compared to spreadsheets.

Question 2.2: Columns

In databases, a "column" refers to a vertical section within a table that stores specific types of data. Each column is assigned a data type, such as text, number, or date, to categorize information consistently. E.g: in a customer database table, a "Phone Number" column could be defined as text to handle alphanumeric phone numbers.

Data types in databases are essential as they specify the kind of data that can be stored in each column, ensuring data accuracy and integrity.

Common Data Types:

1. Text: Used for storing alphanumeric characters such as names, addresses, and descriptions. Example: VARCHAR(50) for storing a maximum of 50 characters of text.
2. Number: Stores numerical values for calculations and statistical analysis. Example: INT for storing whole numbers without decimal places.

3. Date: Stores calendar dates and times for tracking events and scheduling tasks. Example: DATE for storing dates without a specific time component.

Question 2.3: Data Types

1. Importance of Data Types:
   a. Data types are important in databases because they ensure that the data stored is accurate and consistent. They define what kind of data can be stored in each column, preventing errors and maintaining data integrity.
   b. Data types also help in efficient storage by allocating the right amount of space for different types of data, which saves memory and improves database performance.
2. Common Data Types:
   a. Text (VARCHAR or CHAR)**:** Used for storing text data like names, addresses, or descriptions. VARCHAR can vary in length, while CHAR has a fixed length.
   b. Number (INT or DECIMAL): Stores numeric values such as whole numbers (INT) or numbers with decimal points (DECIMAL). They ensure precise calculations and storage of numerical data.
   c. Date (DATE or DATETIME): Stores dates or timestamps. DATE is used for dates only, while DATETIME includes both date and time information. They are crucial for tracking events and scheduling tasks accurately.

Part 3: Expense Tracker Database Design

3.1. Planning

- Expense Amount: How much money was spent for each expense.
- Date: The day when the expense happened.
- Category: What type of expense it was, like "Food", "Transportation", or "Utilities".
- Description/Note: A short explanation of what the expense was for.

- Payment Method: How the expense was paid, such as with cash, credit card, or another method.

3.2. Tables

Columns needed for the "Expenses" table:

1. **expense_id**: INT (Integer), This column stores a unique identifier for each expense. Integer data types (INT) are commonly used for primary keys in databases.
2. **amount**: DECIMAL (Decimal), This column stores the monetary value of each expense. Decimal data types (DECIMAL) are suitable for storing exact values, including cents or fractions of currency.
3. **date**: DATE (Date), This column stores the date when each expense occurred. Date data types (DATE) are used to specifically store calendar dates without any time component.
4. **category**: VARCHAR (Variable-length text), This column categorizes expenses into different types like "Food", "Transportation", etc. VARCHAR data types are used for variable-length text fields, allowing flexibility in category names.
5. **description**: TEXT (Longer text), This column provides a description or note about each expense. TEXT data types are used for storing larger amounts of text, suitable for longer descriptions.
6. **payment_method**: VARCHAR (Variable-length text), This column indicates how each expense was paid, such as "Cash", "Credit Card", etc. VARCHAR data types are used for variable-length text fields to accommodate different payment method names.

Main Table: Expenses

Column Name	Data Type	Description
expense_id	INT	Primary Key, unique identifier.
amount	DECIMAL (10,2)	The amount spent.
date	DATETIME	The date and time of the expense.
category_id	INT	Foreign Key referencing Categories.
payment_method_id	INT	Foreign Key referencing Payment Methods.
description	VARCHAR (250)	A brief description of the expense.
user_id	INT	Foreign Key referencing Users.

Table: Users

Column Name	Data Type	Description
user_id	INT	Primary Key, unique identifier.
user_name	VARCHAR(50)	Name of the user.
email	VARCHAR(100)	Email address of the user.

Table: Categories

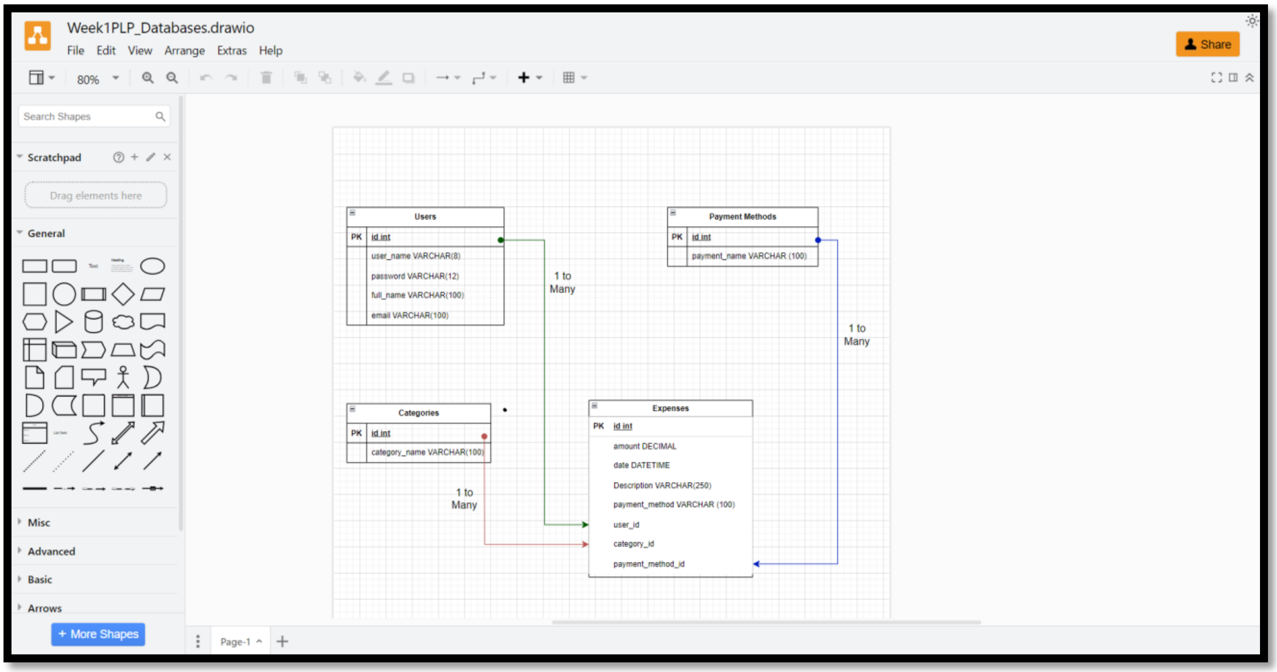
Column Name	Data Type	Description
category_id	INT	Primary Key, unique identifier.
category_name	VARCHAR(100)	Name of the category.

Table: Payment Methods

Column Name	Data Type	Description
payment_method_id	INT	Primary Key, unique identifier.
payment_name	VARCHAR(100)	Name of the payment method.

Written by: Omphemetse Lekaba

Bonus:



Users	
PK	id int
	user_name VARCHAR(8)
	password VARCHAR(12)
	full_name VARCHAR(100)
	email VARCHAR(100)

Payment Methods	
PK	id int
	payment_name VARCHAR (100)

Categories	
PK	id int
	category_name VARCHAR(100)

Expenses	
PK	id int
	amount DECIMAL
	date DATETIME
	Description VARCHAR(250)
	payment_method VARCHAR (100)
	user_id
	category_id
	payment_method_id