

REAL LIFE IMPLEMENTATION OF STRUCTURED QUERY LANGUAGE (SQL)

*A Seminar Report submitted in partial fulfillment of the requirements for the
award of the degree of*

MASTER OF SCIENCE IN MATHEMATICS AND DATA SCIENCE

Submitted by

Ankit Singh

(Roll no. 220227003, Semester III)

Under the supervision of

Prof. Ram Autar

(Head of Department of Mathematics)



DEPARTMENT OF MATHEMATICS

SCHOOL OF BASIC AND APPLIED SCIENCE

**HARCOURT BUTLER TECHNICAL UNIVERSITY
KANPUR, INDIA**

2023 – 2024

ABSTRACT

A Seminar Report for Music Store Analysis involves using the Structured Query Language (SQL) to analyze and derive insights from a database containing information related to a music store. This Report can be a great way to gain valuable insights into the store's operations, Customer Segmentation, trends ,artist effectiveness, artist growth, customer interest and money spent by customers.

Begin by exploring the data to understand its structure and contents. Identify the key tables, columns, and relationships between them. This step is crucial for formulating SQL queries effectively.

This Seminar Report enhances the student's understanding of the theoretical concepts of relational database by providing a simulation of a business environment-like experience. The hands-on practice with SQL and the development of a database which considers the application of referential integrity increases the student's awareness of the complex computer technology requirements within a business organization. This awareness and knowledge prepares the students for successful employment as personnel within the business community.

This report covers the importance of Structured Query language in Real life and presents the working of analyzing data in Database. The report includes the following keypoints:

- Basics of SQL
- About Databases
- Analyzing the Data
- Real life Implementation of SQL

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to the supervisor **Prof. Ram Autar**, for his guidance, constant supervision as well as for providing necessary information regarding the project and also for his support in completing the project. His constant guidance and willingness to share the vast knowledge made me understand this project and its manifestations in great depths and helped me to complete the assigned tasks on time.

I am thankful and fortunate enough to get constant encouragement, support and guidance from all the teaching staff of the Department of Mathematics, which helped me in successfully completing my project work.

Last, but not least, I would like to thank my dear friends, the authors of various research articles and books that were referred to and all those who helped me in one way or another.

Ankit Singh

(Roll no. 220227003)

Department of Mathematics

Harcourt Butler Technical University

Kanpur, Uttar Pradesh

DECLARATION

I Ankit Singh and hereby declare that the work presented in this project dissertation entitled “Real Life implementation of Structured Query Language(SQL)” in partial fulfillment for the award of degree of Master of Science(Mathematics and Data Science) submitted at Department of Mathematics, Harcourt Butler Technical University,Kanpur is an authentic record of work carried out under supervision of Prof. Ram Autar from October 2023 to December 2023.

Date:

Ankit Singh

(Roll no.220227003)

Prof. Ram Autar

Department of Mathematics, HBTU, Kanpur

TABLE OF CONTENTS

ABSTRACT.....	i
ACKNOWLEDGEMENT.....	ii
DECLARATION.....	iii
TABLE OR CONTENTS.....	iv
LIST OF FIGURES.....	v
1 Introduction.....	1
2 History.....	1
3_What is Database?.....	2
4 What is Database Software?.....	3
5 What is Database Management System?.....	3
6 Types of Databases.....	4
7 What is Structured Query Language?.....	11
8 What is a MySQL Database?.....	11
8.1 Syntax.....	12
8.2 Procedural Extensions.....	12
8.3 Reasons for incompatibility.....	12
8.4 Alternatives.....	13
8.5 SQL Data Types.....	13
8.6 Some_of The Most Important SQL Commands.....	14
8.7 What is SQL Used For?.....	14
8.7.1 SQL Used in Marketing.....	15
8.7.2 SQL Used in Healthcare.....	16
9 SQL for Data Analytics.....	18
10 SQL for Data Science.....	18
11 Why should i learn SQL for Data Science.....	19
12 Statements used in SQL.....	20
13 Filtering Data in SQL.....	20

14 SQL JOINS.....	21
14.1 Inner Joins.....	21
14.2 Outer Joins.....	22
14.3 Cross Joins.....	23
15 Real Life Implementation in SQL.....	24
16 Digital Music Store Analysis.....	25
17 CONS.....	28
18 PROS.....	29
19 TOOLS.....	30
20 CONCLUSION.....	31
REFERENCE.....	33

REAL LIFE IMPLEMENTATION OF STRUCTURED QUERY LANGUAGE

1 INTRODUCTION

is a domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS). It is particularly useful in handling structured data, i.e., data incorporating relations among entities and variables.

Introduced in the 1970s, SQL offered two main advantages over older read–write APIs such as ISAM or VSAM. Firstly, it introduced the concept of accessing many records with one single command. Secondly, it eliminates the need to specify *how* to reach a record, i.e., with or without an index.

Originally based upon relational algebra and tuple relational calculus, SQL consists of many types of statements, which may be informally classed as sublanguages, commonly: a data query language (DQL), a data definition language (DDL), a data control language (DCL), and a data manipulation language (DML). The scope of SQL includes data query, data manipulation (insert, update, and delete), data definition (schema creation and modification), and data access control. Although SQL is essentially a declarative language (4GL), it also includes procedural elements.

SQL was one of the first commercial languages to use Edgar F. Codd's relational model. The model was described in his influential 1970 paper, "A Relational Model of Data for Large Shared Data Banks". Despite not entirely adhering to the relational model as described by Codd, SQL became the most widely used database language.

SQL became a standard of the American National Standards Institute (ANSI) in 1986 and of the International Organization for Standardization (ISO) in 1987. Since then, the standard has been revised multiple times to include a larger set of features and incorporate common extensions. Despite the existence of standards, virtually no implementations in existence adhere to it fully, and most SQL code requires at least some changes before being ported to different database systems.

2 HISTORY

SQL was initially developed at IBM by Donald D. Chamberlin and Raymond F. Boyce after learning about the relational model from Edgar F. Codd in the early 1970s. This version, initially called SEQUEL (Structured English QUery Language), was designed to manipulate and retrieve data stored in IBM's original quasi relational database management system, System R, which a group at IBM San Jose Research Laboratory had developed during the 1970s.

Chamberlin and Boyce's first attempt at a relational database language was SQUARE (Specifying Queries in A Relational Environment), but it was difficult to use due to subscript/superscript notation. After moving to the San Jose Research Laboratory in 1973, they began work on a sequel to SQUARE. The original name SEQUEL, which is widely regarded as a pun on QUEL, the query language of Ingres, was later changed to SQL (dropping the vowels) because "SEQUEL" was a trademark of the UK-based Hawker Siddeley Dynamics Engineering Limited company. The label SQL later became the acronym for Structured Query Language.

After testing SQL at customer test sites to determine the usefulness and practicality of the system, IBM began developing commercial products based on their System R prototype, including System/38, SQL/DS, and IBM Db2, which were commercially available in 1979, 1981, and 1983, respectively.

In the late 1970s, Relational Software, Inc. (now Oracle Corporation) saw the potential of the concepts described by Codd, Chamberlin, and Boyce, and developed their own SQL-based RDBMS with aspirations of selling it to the U.S. Navy, Central Intelligence Agency, and other U.S. government agencies. In June 1979, Relational Software introduced one of the first commercially available implementations of SQL, Oracle V2 (Version2) for VAX computers.

By 1986, ANSI and ISO standard groups officially adopted the standard "Database Language SQL" language definition. New versions of the standard were published in 1989, 1992, 1996, 1999, 2003, 2006, 2008, 2011, 2016 and most recently, 2023.

3 WHAT IS DATABASE ?

- Database is an organized collection of structured information or data stored electronically in a computer system.
- It's designed to efficiently manage, store, retrieve, and manipulate data according to predefined requirements.
- Databases serve as repositories for various types of information, allowing users and applications to access, update, and manage data.
- A database is usually controlled by a database management system (DBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just a database.
- Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient.
- The data can then be easily accessed, managed, modified, updated, controlled, and organized. Most databases use structured query language (SQL) for writing and querying data.

3.1 What is Database Software?

Database software is used to create, edit, and maintain database files and records, enabling easier file and record creation, data entry, data editing, updating, and reporting. The software also handles data storage, backup and reporting, multi-access control, and security. Strong database security is especially important today, as data theft becomes more frequent. Database software is sometimes also referred to as a “database management system” (DBMS).

Database software makes data management simpler by enabling users to store data in a structured form and then access it. It typically has a graphical interface to help create and manage the data and, in some cases, users can construct their own databases by using database software.

3.2 What is a database management system (DBMS)?

A database typically requires a comprehensive database software program known as a database management system (DBMS). A DBMS serves as an interface between the database and its end users or programs, allowing users to retrieve, update, and manage how the information is organized and optimized. A DBMS also facilitates oversight and control of databases, enabling a variety of administrative operations such as performance monitoring, tuning, and backup and recovery.

Some examples of popular database software or DBMSs include MySQL, Microsoft Access, Microsoft SQL Server, FileMaker Pro, Oracle Database, and dBASE.

3.3 Types of databases

There are many different types of databases. The best database for a specific organization depends on how the organization intends to use the data.

Relational database : became dominant in the 1980s. Items in a relational database are organized as a set of tables with columns and rows. Relational database technology provides the most efficient and flexible way to access structured information.

Object-oriented databases: Information in an object-oriented database is represented in the form of objects, as in object-oriented programming.

Distributed databases: A distributed database consists of two or more files located in different sites. The database may be stored on multiple computers, located in the same physical location, or scattered over different networks.

Data warehouses: A central repository for data, a data warehouse is a type of database specifically designed for fast query and analysis.

NoSQL databases: A NoSQL, or non relational database, allows unstructured and semistructured data to be stored and manipulated (in contrast to a relational database, which defines how all data inserted into the database must be composed). NoSQL databases grew popular as web applications became more common and more complex.

Graph databases: A graph database stores data in terms of entities and the relationships between entities.

OLTP databases: An OLTP database is a speedy, analytic database designed for large numbers of transactions performed by multiple users.

These are only a few of the several dozen types of databases in use today. Other, less common databases are tailored to very specific scientific, financial, or other functions. In addition to the different database types, changes in technology development approaches and dramatic advances such as the cloud and automation are propelling databases in entirely new directions. Some of the latest databases include

Open source databases: An open source database system is one whose source code is open source; such databases could be SQL or NoSQL databases.

Cloud databases: A cloud database is a collection of data, either structured or unstructured, that resides on a private, public, or hybrid cloud computing platform. There are two types of cloud database models: traditional and database as a service (DBaaS). With DBaaS, administrative tasks and maintenance are performed by a service provider.

Multimodal database: Multimodel databases combine different types of database models into a single, integrated back end. This means they can accommodate various data types.

Document/JSON database Designed for storing, retrieving, and managing document-oriented information, document databases are a modern way to store data in JSON format rather than rows and columns.

Self-driving databases The newest and most groundbreaking type of database, self-driving databases (also known as autonomous databases) are cloud-based and use machine learning to automate database tuning, security, backups, updates, and other routine management tasks traditionally performed by database administrators.

3.4 What is Structured Query Language (SQL)?

SQL is a programming language used by nearly all relational databases to query, manipulate, and define data, and to provide access control. SQL was first developed at IBM in the 1970s with Oracle as a major contributor, which led to implementation of the SQL ANSI standard, SQL has spurred many extensions from companies such as IBM, Oracle, and Microsoft. Although SQL is still widely used today, new programming languages are beginning to appear.

3.5 What is a MySQL database?

MySQL is an open source relational database management system based on SQL. It was designed and optimized for web applications and can run on any platform. As new and different requirements emerged with the internet, MySQL became the platform of choice for web developers and web-based applications. Because it's designed to process millions of queries and thousands of transactions, MySQL is a popular choice for ecommerce businesses that need to manage multiple money transfers. On-demand flexibility is the primary feature of MySQL.

MySQL is the DBMS behind some of the top websites and web-based applications in the world, including Airbnb, Uber, LinkedIn, Facebook, Twitter, and YouTube.

4 SYNTAX

The SQL language is subdivided into several language elements, including:

- Clauses, which are constituent components of statements and queries. (In some cases, these are optional.)
- Expressions, which can produce either scalar values, or tables consisting of columns and rows of data
- Predicates, which specify conditions that can be evaluated to SQL three-valued logic (3VL) (true/false/unknown) or Boolean truth values and are used to limit the effects of statements and queries, or to change program flow.
- Queries, which retrieve the data based on specific criteria. This is an important element of SQL.
- Statements, which may have a persistent effect on schemata and data, or may control transactions, program flow, connections, sessions, or diagnostics.
- SQL statements also include the semicolon (";") statement terminator. Though not required on every platform, it is defined as a standard part of the SQL grammar.
- Insignificant whitespace is generally ignored in SQL statements and queries, making it easier to format SQL code for readability.

```
UPDATE clause {UPDATE country
SET clause   {SET population = population + 1
WHERE clause {WHERE name = 'USA';
```

expression
expression
predicate

statement

5 PROCEDURAL EXTENSIONS

SQL is designed for a specific purpose: to query data contained in a relational database. SQL is a set-based, declarative programming language, not an imperative programming language like C or BASIC. However, extensions to Standard SQL add procedural programming language functionality, such as control-of-flow constructs.

In addition to the standard SQL/PSM extensions and proprietary SQL extensions, procedural and object-oriented programmability is available on many SQL platforms via DBMS integration with other languages. The SQL standard defines SQL/JRT extensions (SQL Routines and Types for the Java Programming Language) to support Java code in SQL databases. Microsoft SQL Server 2005 uses the SQLCLR (SQL Server Common Language Runtime) to host managed .NET assemblies in the database, while prior versions of SQL Server were restricted to unmanaged extended stored procedures primarily written in C. PostgreSQL lets users write functions in a wide variety of languages—including Perl, Python, Tcl, JavaScript (PL/V8) and C.

6 Reasons for incompatibility

Several reasons for the lack of portability between database systems include:

- The complexity and size of the SQL standard means that most implementers do not support

the entire standard.

- The SQL standard does not specify the database behavior in some important areas (e.g., indices, file storage), leaving implementations to decide how to behave.
- The SQL standard defers some decisions to individual implementations, such as how to name a results column that was not named explicitly.^{[27]:207}
- The SQL standard precisely specifies the syntax that a conforming database system must implement. However, the standard's specification of the semantics of language constructs is less well-defined, leading to ambiguity.
- Many database vendors have large existing customer bases; where the newer version of the SQL standard conflicts with the prior behavior of the vendor's database, the vendor may be unwilling to break backward compatibility.
- Little commercial incentive exists for vendors to make changing database suppliers easier (see vendor lock-in).
- Users evaluating database software tend to place other factors such as performance higher in their priorities than standards conformance.

7 Alternatives

A distinction should be made between alternatives to SQL as a language, and alternatives to the relational model itself. Below are proposed relational alternatives to the SQL language. See navigational database and NoSQL for alternatives to the relational model.

- .QL: object-oriented Datalog
- 4D Query Language (4D QL)
- Datalog: critics suggest that Datalog has two advantages over SQL: it has cleaner semantics, which facilitates program understanding and maintenance, and it is more expressive, in particular for recursive queries.^[40]
- HTSQL: URL based query method
- IBM Business System 12 (IBM BS12): one of the first fully relational database management systems, introduced in 1982
- ISBL
- jOOQ: SQL implemented in Java as an internal domain-specific language
- Java Persistence Query Language (JPQL): The query language used by the Java Persistence API and Hibernate persistence library
- JavaScript: MongoDB implements its query language in a JavaScript API.
- LINQ: Runs SQL statements written like language constructs to query collections directly from inside .Net code
- Object Query Language
- QBE (Query By Example) created by Moshè Zloof, IBM 197
- QUEL introduced in 1974 by the U.C. Berkeley Ingres project, closer to tuple relational calculus than SQL
- XQuery

8 SQL Data Types

The SQL standard defines three kinds of data types :

- predefined data types
- constructed types

- user-defined types.

Constructed types are one of ARRAY, MULTISSET, REF(erence), or ROW. *User-defined types* are comparable to classes in object-oriented language with their own constructors, observers, mutators, methods, inheritance, overloading, overwriting, interfaces, and so on. *Predefined data types* are intrinsically supported by the implementation

8.1 Some of The Most Important SQL Commands

- **SELECT** - extracts data from a database
- **UPDATE** - updates data in a database
- **DELETE** - deletes data from a database
- **INSERT INTO** - inserts new data into a database
- **CREATE DATABASE** - creates a new database
- **ALTER DATABASE** - modifies a database
- **CREATE TABLE** - creates a new table
- **ALTER TABLE** - modifies a table
- **DROP TABLE** - deletes a table
- **CREATE INDEX** - creates an index (search key)
- **DROP INDEX** - deletes an index

8.2 Characteristics and Benefits of Structured Query Language:

The ANSI SQL provides with:

- Specific syntax and semantics of SQL data definition and data manipulation languages.
- It also provides with basic data structure and operations for designing, assessing, maintaining, controlling and protecting SQL databases.
- Portability of database definition and application is also provided. Applications can be moved from one machine to another.
- IS professionals share a common language and reduce training costs.
- Professionals can become proficient in its use and increase the

productivity.

- It provides with longevity.
- It provides with reduced dependence on single vendor.

8.3 What is SQL Used For?

Industries That Use SQL

1. SQL Uses in Marketing

- Marketing teams often target customers and release promotions based on user data collected by the organization.
- More often than not, this data is stored in large databases and must be queried before marketers can use it.
- Here is an example of how SQL is used in the field of marketing:
- Anne is a marketing executive at an ecommerce company. Every time users interact with the company's website, their data is collected and automatically stored in a relational database. There are over 1.5 million records of user data stored in the database.
- Variables collected include items purchased by the customer, their gender,

time of purchase, and a unique user ID.

Here is a summarized view of the database table she needs to query

Customer ID	Gender	Item Purchased	Date
1001	Female	Floral Black Dress	2021-08-15
1002	Female	Sleeveless Zipper Jumpsuit	2021-08-17
1003	Male	Graphic Thermal Hoodie	2021-08-25

From the table above, Anne needs to extract all the details of customers who made purchases during a sale on 25th August 2021. She wants to retarget these users for a similar marketing campaign she is running next quarter.

Here is a simple SQL query she can write to access the data she needs in less than a minute:

```
SELECT * FROM customer_data A WHERE CAST (A.Date AS Date) >= '2021-08-25';
```

This query will return the following results from the table displayed above:

Customer ID	Gender	Item Purchased	Date
1003	Male	Graphic Thermal Hoodie	2021-08-25

The example above demonstrates just how useful SQL is in extracting relevant information analyze business data.

Since the marketing process is becoming increasingly data-driven, it is a good idea for marketers to learn basic SQL so that they can make informed decisions when running campaigns and targeting users

1.2 SQL Uses in Healthcare

SQL is often used to manipulate and analyze patient information stored in clinical databases. For example, it can be used to build dashboards on user health data, generate patient reports based on input from medical professionals, and even create searches against large databases like the Covid-19 tracker.

Here is an example of how SQL is used in the healthcare domain:

A non-profit charitable organization collects data related to a deadly disease that is prevalent in rural areas. Volunteers then visit high-risk areas with the largest number of infections to administer vaccines to curb the spread of the disease.

Here are five rows of a sample database created by the organization:

This is an SQL query that can be written to identify the top 10 regions with the highest number of infections:

SELECT * from diseases

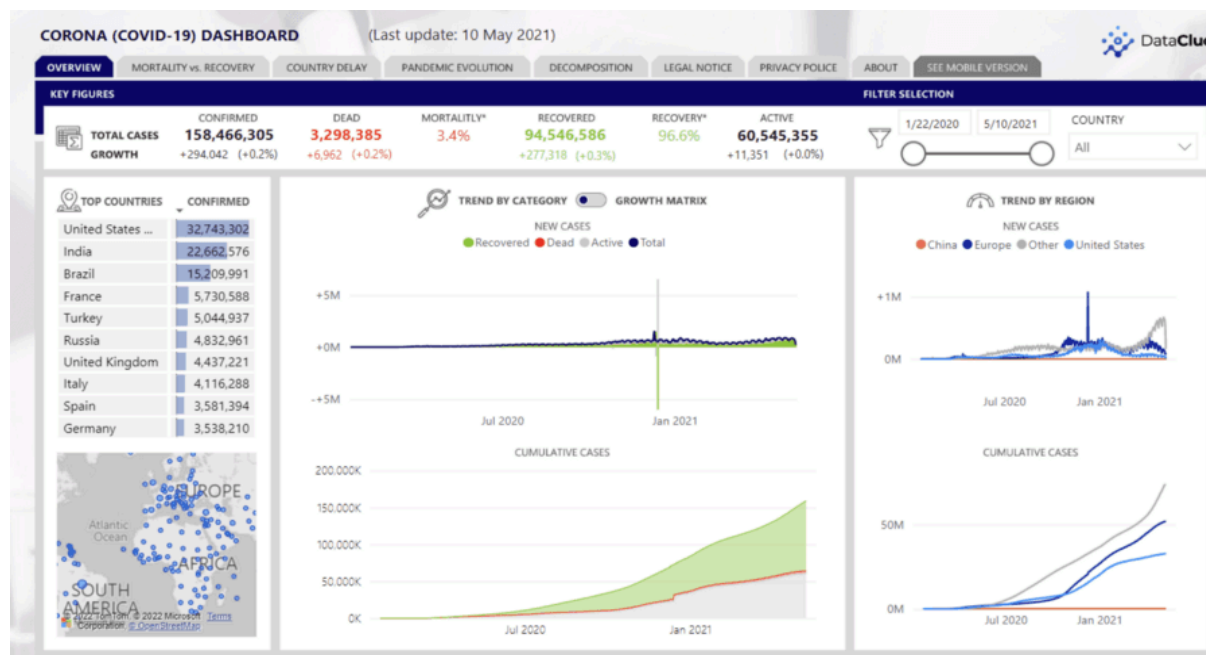
ORDER BY Cases DESC

LIMIT 10

Country	Cases
Afghanistan	55
Africa	21
Albania	1
Algeria	0
Americas	0

While the query above is useful, it is inefficient for volunteers to run it on a daily basis to find high-risk regions. Instead, the SQL database can be connected to a live, real-time dashboard that visualizes the number of infections in different regions.

Here is a sample Covid-19 dashboard that does just that:



8.4 SQL For Data Analytics

Data analysts are skilled professionals who identify trends in data to aid in the company's decision-making process. They use SQL for data extraction and analysis to generate actionable insights.

Managers and stakeholders use the patterns uncovered by data analysts to make decisions that add business value to the organization.

Data analysts are some of the most sought-after professionals in the world. Take the **Data Analyst in SQL** career track to begin your transition into the field today.

8.5 SQL For Data Science

Data science involves extracting insights from data, and SQL is key for this, especially in predictive modeling and analysis. SQL is crucial for:

1. **Data Extraction and Preprocessing:** Vital for querying and preparing data from databases, enabling filtering, sorting, and aggregation for high-quality data sets.
2. **Exploratory Data Analysis (EDA):** Helps data scientists understand patterns and correlations in databases, aiding in hypothesis formation and modeling approaches.
3. **Feature Engineering:** Used to create new variables from existing data to enhance model performance, utilizing SQL's functions and table joining capabilities.
4. **Data Wrangling for Machine Learning:** Transforms and structures data for machine learning, dealing with normalization, missing values, and categorical encoding.
5. **Integration with Analytical Tools:** Seamlessly integrates with tools and languages like Python and R, combining SQL's data manipulation with advanced analytics.
6. **Scalability and Performance:** Handles large data volumes efficiently, crucial for time and resource-limited environments.
7. **Reporting and Visualization:** Generates datasets for visualization tools and reporting software, ensuring accurate and up-to-date reports.
8. **Real-time Data Science:** Quick querying and processing abilities are vital for real-time applications like dynamic pricing, fraud detection, and recommendation systems.

8.6 Why Should You Learn SQL for Data Science?

When most people think of data science, their minds often jump to predictive analytics and machine learning modeling. However, you can only build machine learning models if you have the data necessary to do so. In the real world, this data will rarely be handed to you in a clean, structured spreadsheet like it is on Kaggle.

You need to access the company's database, collect the data you need, and pre-process it

before you can start building predictive models. As mentioned above, most companies store data in relational databases, meaning you need to know SQL to achieve this.

In fact, according to a recent study, SQL is a requirement in almost 65% of all data science job listings. This means that even if you know Python, you are missing out on around 3 out of 5 job opportunities if you lack SQL skills.

8.7 STATEMENTS USED IN SQL

- **SELECT Statement:** Retrieve Data from a database.

SYNTAX: SELECT column1, column2, ... FROM table_name WHERE condition;

- **INSERT INTO Statement:** Add a new record to the table.

SYNTAX: INSERT INTO table_name (column1, column2, ...) VALUES (value1, value2, ...);

- **UPDATE Statement:** Modifying existing record in the table.

SYNTAX: UPDATE table_name SET column1 = value1, column2 = value2, ... WHERE condition;

8.8 FILTERING DATA USED IN SQL

WHERE Clause: Filter records returned by
'SELECT', 'UPDATE', 'DELETE' Statement.

SYNTAX: SELECT column1, column2, ... FROM table_name WHERE condition;

LOGICAL Operators: SQL provides logical operators such as AND, OR, and NOT to combine multiple conditions for more complex data filtering.

SYNTAX: SELECT * FROM table_name WHERE condition1 AND condition2;

8.9 SQL JOIN

A **JOIN** clause is used to combine rows from two or more tables, based on a related column between them.

Let's look at a selection from the "Orders" table:

OrderID	CustomerID	OrderDate
10308	2	1996-09-18
10309	37	1996-09-19
10310	77	1996-09-20

Then, look at a selection from the "Customers" table:

CustomerID	CustomerName	ContactName	Country
1	Alfreds Futterkiste	Maria Anders	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Mexico

Notice that the "CustomerID" column in the "Orders" table refers to the "CustomerID" in the "Customers" table. The relationship between the two tables above is the "CustomerID" column.

Then, we can create the following SQL statement (that contains an **INNER JOIN**), that selects records that have matching values in both tables:

Example

```
SELECT Orders.OrderID, Customers.CustomerName,  
       Orders.OrderDate
```

```
FROM Orders
```

```
INNER JOIN Customers ON  
       Orders.CustomerID=Customers.CustomerID;
```

and it will produce something like this:

OrderID	CustomerName	OrderDate
10308	Ana Trujillo Emparedados y helados	9/18/1996
10365	Antonio Moreno Taquería	11/27/1996
10383	Around the Horn	12/16/1996

10355 Around the Horn

11/15/1996
6

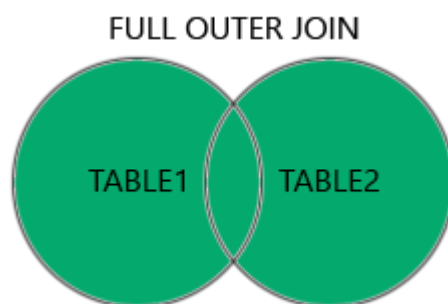
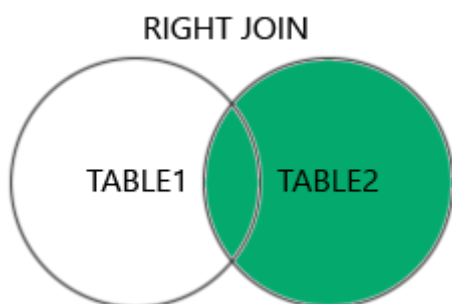
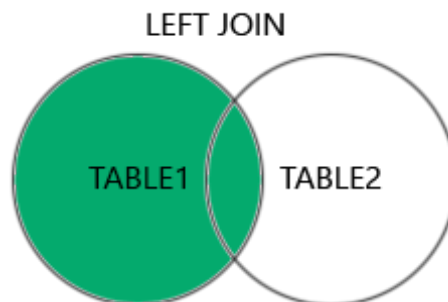
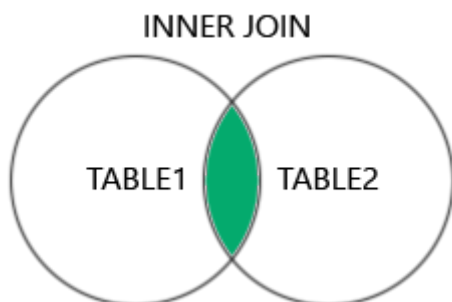
10278 Berglunds snabbköp

8/12/1996

8.10 Different Types of SQL JOINS

Here are the different types of the JOINS in SQL:

- **(INNER) JOIN**: Returns records that have matching values in both tables
- **LEFT (OUTER) JOIN**: Returns all records from the left table, and the matched records from the right table
- **RIGHT (OUTER) JOIN**: Returns all records from the right table, and the matched records from the left table
- **FULL (OUTER) JOIN**: Returns all records when there is a match in either left or right table

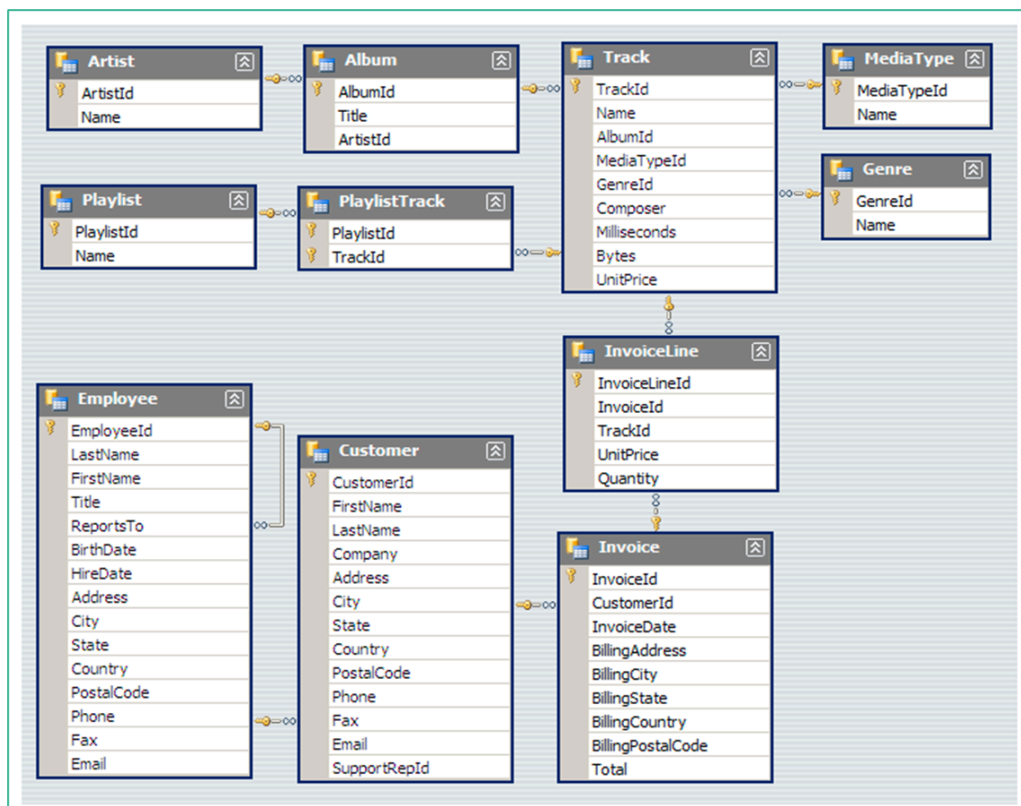


Digital Music Store Analysis :

Analyzing a music store database involves extracting insights from the data it contains.

1. Understanding the Database Schema:

- Review the structure of the music store database. Understand the tables, relationships, and the data they store.

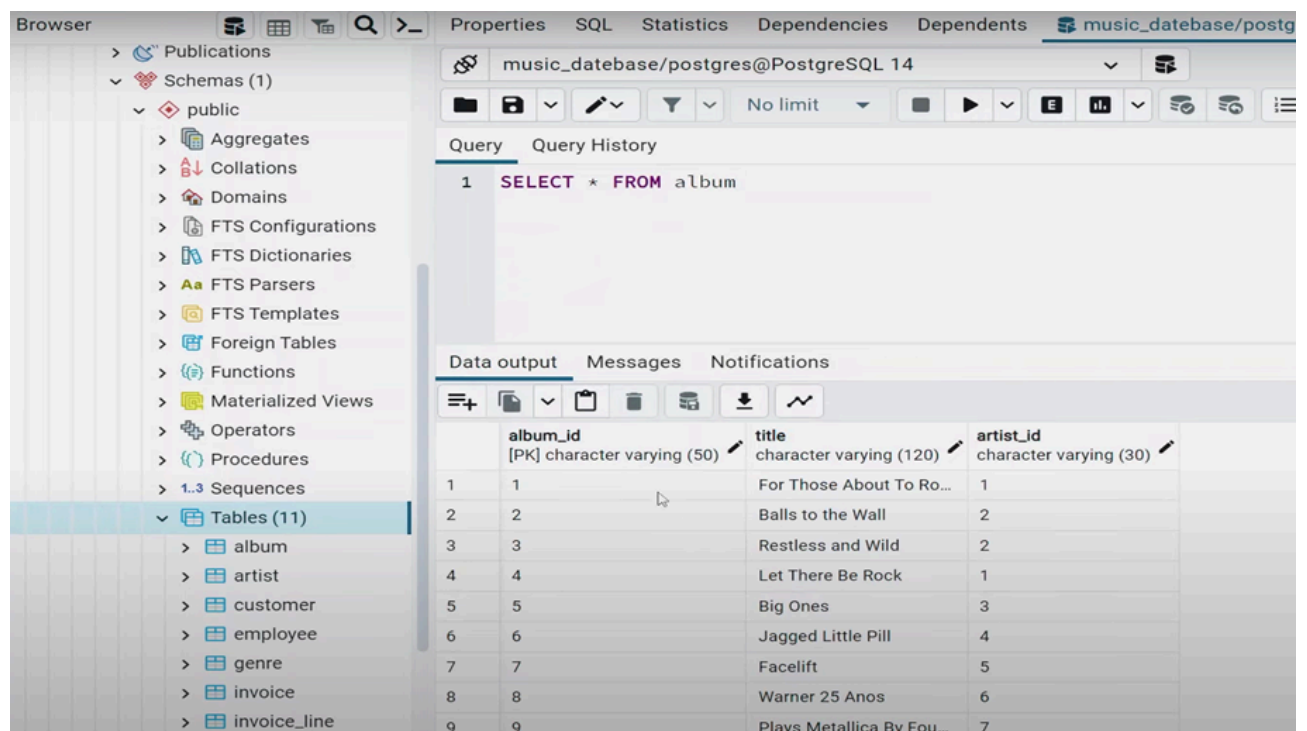


2. Connecting to the Database:

Use a SQL client or command-line interface to connect to the music store database.

3. Exploratory Queries:

- Write basic queries to explore the data.



The screenshot shows a PostgreSQL database client interface. On the left, a 'Browser' pane displays the database structure, including 'public' schema, 'Aggregates', 'Collations', 'Domains', 'FTS Configurations', 'FTS Dictionaries', 'FTS Parsers', 'FTS Templates', 'Foreign Tables', 'Functions', 'Materialized Views', 'Operators', 'Procedures', 'Sequences', and 'Tables (11)'. The 'Tables (11)' section is expanded, showing a list of tables: 'album', 'artist', 'customer', 'employee', 'genre', 'invoice', and 'invoice_line'. The 'album' table is selected. The main pane displays the query 'SELECT * FROM album' and its results. The 'Data output' tab is active, showing a table with three columns: 'album_id' (PK, character varying (50)), 'title' (character varying (120)), and 'artist_id' (character varying (30)). The results are as follows:

album_id	title	artist_id
1	For Those About To Ro...	1
2	Balls to the Wall	2
3	Restless and Wild	2
4	Let There Be Rock	1
5	Big Ones	3
6	Jagged Little Pill	4
7	Facelift	5
8	Warner 25 Anos	6
9	Plays Metallica By Fou...	7

4. Data Cleaning and Transformation:

- Identify any inconsistencies or missing data.
- Clean the data by removing duplicates, handling null values, or standardizing formats.

5. Performing Analysis:

SQL PROJECT- MUSIC STORE DATA ANALYSIS

Question Set 1 - Easy

1. Who is the senior most employee based on job title?
2. Which countries have the most Invoices?
3. What are top 3 values of total invoice?
4. Which city has the best customers? We would like to throw a promotional Music Festival in the city we made the most money. Write a query that returns one city that has the highest sum of invoice totals. Return both the city name & sum of all invoice totals
5. Who is the best customer? The customer who has spent the most money will be declared the best customer. Write a query that returns the person who has spent the most money

Question Set 2 – Moderate

1. Write query to return the email, first name, last name, & Genre of all Rock Music listeners. Return your list ordered alphabetically by email starting with A
2. Let's invite the artists who have written the most rock music in our dataset. Write a query that returns the Artist name and total track count of the top 10 rock bands

/* Q1: Who is the senior most employee based on job title? */

SELECT title, last_name, first_name

FROM employee

ORDER BY levels DESC

LIMIT 1

/* Q2: Which countries have the most Invoices? */

SELECT COUNT(*) AS c, billing_country

FROM invoice

GROUP BY billing_country

ORDER BY c DESC

/* Q3: What are the top 3 values of total invoice? */

SELECT total

FROM invoice

ORDER BY total DESC

PROS:

1.**Ease of Use:** SQL has a relatively simple syntax, making it easy to learn and use, especially for querying and managing relational databases.

2.**Standardization:** Being a standardized language, SQL allows for portability across various database platforms, making it easier to transfer skills and code between systems.

3.**Robustness:** It can handle complex queries and database operations, supporting a wide range of functionalities like data retrieval, manipulation, and administration.

4.**Scalability:** SQL databases can scale vertically by adding more resources to a single server or horizontally by distributing data across multiple servers.

5.**Data Integrity:** SQL supports constraints (e.g., foreign keys, unique keys) ensuring data integrity, reducing the risk of inconsistent or incorrect data.

6.**Security:** SQL databases provide strong security features, including user access controls, encryption, and authentication mechanisms, safeguarding sensitive data.

CONS:

1. **Limited Support for Non-Relational Data:** SQL is primarily designed for relational databases, which might not be the ideal choice for handling non-relational, unstructured, or hierarchical data.
2. **Performance Bottlenecks:** Poorly optimized queries or database design can lead to performance issues, especially with large datasets or complex queries.
3. **Learning Curve:** While basic SQL is easy to learn, mastering advanced concepts, optimization techniques, and best practices might require significant time and experience.
4. **Vendor Lock-in:** Different database vendors might have proprietary extensions or implementations, potentially leading to vendor lock-in for advanced functionalities.
5. **Complex Joins:** Complex joins across multiple tables can be challenging to write, understand, and optimize, impacting query performance.
6. **Scaling Challenges:** While SQL databases can scale, horizontal scaling (across multiple servers) can be more complex compared to some NoSQL databases designed for this purpose.

TOOLS

SQL Query Development and Management:

1.**SQL Server Management Studio (SSMS):** Microsoft's tool for managing SQL Server databases, offering a rich environment for writing queries, managing databases, and performing administrative tasks.

2.**MySQL Workbench:** An official tool for MySQL, providing SQL development, database administration, and data modelling capabilities.

3.**Oracle SQL Developer:** Oracle's tool for developing and managing Oracle databases, featuring SQL development, query optimization, and database administration .

Data Visualization and Business Intelligence

1. **Tableau:** A powerful BI tool that can connect to SQL databases for data visualization, dashboards, and analytics.
2. **Power BI:** Microsoft's business analytics tool supporting SQL databases, allowing data visualization, reporting, and dashboard creation.
3. **Looker:** A data exploration and visualization platform that can connect to SQL databases for analyzing and visualizing data.

10 Conclusion

In conclusion, we've explored the practical implementation of SQL in various aspects of data management, from database design to query optimization. Let's recap the key takeaways:

Database Design and Schema Definition:

Proper database design is fundamental for efficient data storage.

SQL allows us to define clear schemas, ensuring data integrity and organization.

Data Retrieval and Manipulation:

SQL's SELECT statement is a powerful tool for retrieving specific data from databases.

We've seen how to filter, sort, and aggregate data for meaningful insights.

Advanced Query Techniques:

Advanced SQL techniques, including subqueries and JOIN operations, enhance our ability to retrieve complex datasets.

Optimization strategies such as indexing contribute to improved query performance.

Real-World Applications:

Through practical examples, we've applied SQL to real-world scenarios.

Analyzing sales data, understanding customer behavior, and visualizing results showcase SQL's versatility.

Challenges and Solutions:

We've discussed common challenges in SQL implementation, such as performance bottlenecks.

Strategies for optimization and best practices help mitigate these challenges.

Implications for Business and Development:

The real implementation of SQL extends beyond the technical realm. It directly impacts businesses and development processes:

Business Intelligence: SQL enables organizations to extract valuable insights from their data, supporting informed decision-making.

Scalability: As businesses grow, SQL databases can scale vertically or horizontally to handle increased data volumes.

Application Development: SQL is at the core of many applications, ensuring seamless interaction with databases for dynamic and responsive systems.

Continued Learning and Exploration:

As we conclude, remember that SQL is a vast and continually evolving field. Embrace a mindset of continuous learning:

Stay updated with SQL advancements and best practices.

Practice writing and optimizing queries to enhance your skills.

By applying the principles we've covered today, you can leverage SQL's power to its fullest potential in your professional journey.

Thank you for your participation and engagement! Any questions or thoughts? Let's open the floor for a final discussion.

REFERENCES

<https://en.wikipedia.org/wiki/SQL>

<https://www.oracle.com/in/database/what-is-database/>

<https://www.datacamp.com/blog/what-is-sql-used-for>

[4] Yilmaz A, Li X, Shah M. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. IEEE Transactions on Pattern Analysis and Machine Intelligence.2004;26(11):1531-1536

[6] Lim J, Ross DA ,Lin RS , Yang MH. Incremental learning for visual tracking . Advances in Neural Information Processing Systems, 2004:793-800

[7] Wang F, Lu M. Hamiltonian Monte Carlo estimator for abrupt motion tracking. In. International Conference on Pattern Recognition ,ICPR. Nov.2012 p.p 3066-3069

[8] Zhang H, Zhang J, Wu Q, Qian X, Zhou T, Hengcheng FU. Extended Kernel correlation filter for abrupt motion tracking. KSII Transactions on Internet and Information Systems.2017;11(9):4438-4460

[11] Zhou X, Lu Y, Lu J, Zhou J. Abrupt motion tracking via intensively adaptive Markov-chain Monte Carlo sampling. IEEE Transactions on Image Processing. 2012;21(2):789-801

[12] Balan A, Black MJ. An adaptive appearance model approach for model based articulated object tracking. In IEEE Conference Computer Vision and Pattern recognition ,June 2006

- [13] Porikli F, Tuzel O. Covariance tracking using model update based on lie algebra , In: IEEE Computer Society Conference on Computer Vision and Pattern recognition , CVPR; 1. June 2006, pp. 728-7
- [14] Hou L, Wan W, Lee KH, Hwang JN, Okopal G, Pitton J. Robust human tracking based on DPM constrained multiple-kernels from moving cameras. Journal of sigma processing systems
- [15] Canyameres Masip , Sergi, and Antonio Manuel Lopez Pea *On the use of Convolution Neural Networks for Pedestrian Detection* .2015
- [16] Szarvas, M., Yoshizawa , A., Yamamoto, M., and gata, J. *Pedestrian detection with convolution neural networks*. Intelligent Vehicles Symposium.2005
- [17] P. Dollar ,C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art." IEEE transactions on pattern analysis and machine intelligence,34(4), 743-761,2012
- [18] N. Dalal, and B. trigs, "Histograms of oriented gradients for human detection," In Computer Vision and Pattern Recognition, IEEE Computer Society Conference on Vol.1,pp.886-893,2005
- [19] M. Bertozzi, A. Broggi, P. Grisleri, T. graf, M. Meinecke,"Pedestrian