**Dataset - Importance of having a schema in datasets(Primary key)**

**Introduction**

**Overview of Data Types (Structured)**

Data types are fundamental building blocks in data management, defining the nature of the data that can be stored, processed, and analyzed. Structured data types are organized and easily searchable, typically residing in relational databases or spreadsheets where they follow a predefined format. Common structured data types include:

* **Integer**: Whole numbers, used for counting or indexing.
* **Float**: Decimal numbers, representing continuous values.
* **String**: Sequences of characters, often used for text or alphanumeric codes.
* **Date/Time**: Specific points in time, useful for tracking events or scheduling.

**Importance of Schema in Data Management**

A schema is a blueprint or structure that defines how data is organized within a database. It specifies the tables, fields, data types, and relationships among the data elements, serving as a crucial aspect of data management. The importance of schema lies in several key areas:

* **Data Integrity**: By enforcing data types and constraints, a schema ensures that data entered into a database adheres to predefined rules, preventing errors and inconsistencies.
* **Efficient Querying**: Schemas enable the efficient organization of data, allowing for faster and more effective querying and retrieval of information.
* **Scalability**: A well-designed schema facilitates the expansion and scaling of databases, supporting the addition of new data types or relationships without disrupting existing structures.
* **Documentation and Maintenance**: A schema serves as documentation of the database structure, making it easier for developers and data managers to maintain and update the database over time.

**DATA SET 1 – NETFLIX\_TITLES**

*This tabular dataset consists of listings of all the movies and tv shows available on Netflix, along with other details. They have 8800 row of movie and shows listings.*

* We can analyze number of movies/TV shows released in each year.
* We can analyze number of seasons each show has.
* We can categorize movies/tv shows on basis of the ratings, type.
* We can check how many movies/ shows a director release in Netflix and on which years.
* Understanding what content is available in different countries.
* Movies vs shows

**Source :** <https://www.kaggle.com/datasets/shivamb/netflix-shows>

A screenshot of a computer

Description automatically generated

**TABLE DETAILS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data\_Type** | **Description** | **Observation** |
| Show\_ID | String | Each show is given an ID | This has Unique value S1 to S8087. Primary key |
| Type | String | Movie/ TV Show |  |
| Title | String | Title of the movie | It has duplicates |
| Director | String | Name of the director | This has Null values. Would be difficult to group since spacing error |
| Cast | String | Cast of the crew | All the names in one cell. Difficult to query |
| Country | String | Based on which country | This has Null values. |
| Date of Release | String | Release date | Would be easier if it is in DATE datatype |
| Release year | integer | Year of release |  |
| Rating | String | 17 Rating types |  |
| Duration | String | Duration of movie | Some are in Mins, some are in seasons. Difficult to query |
| Listed\_in | String | Type of the movie/show. Drama, Romace Horror, Comedy, Thriller…etc | All the listings in one cell. |
| Description | string | Storyline | Can be DELETED |

**To find duplicates/ Null values :**

**A screenshot of a computer

Description automatically generated**

**Advantages:**

* Easy to query
* Only one table is available in the data set. No need to use complex queries like JOIN
* Primary key availability makes easy to uniquely identify each data.

**Disadvantages**

* The data is not consistent. There is no parent child relationship
* It is a very big data and raw data. We can expect spelling mistakes which makes analyzing difficult.
* Null values will affect the end result.

**DATA SET 2 – COFFEE SHOP SALES**

*This tabular dataset consists of listings of all the transactions happened from Jan to June and the product name, along with other details. They have 149456 rows of the transactions*

* We can analyze transaction by product type and the revenue it generated
* We can analyze monthly revenue generated.
* We can check which day/hour is the busiest
* We can analyze all the above based on Store location, product category, everyday sales
* We can also analyze how a product price affects the sales

**Source :** <https://www.kaggle.com/datasets/ahmedmohamedibrahim1/coffee-shop-sales-dataset>

**A screenshot of a computer

Description automatically generated**

**TABLE DETAILS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data\_Type** | **Description** | **Observation** |
| Transaction\_ID | Integer | Each transaction is given an ID | This has Unique value 1 to 149456. Primary key |
| Transaction\_date | Date | Date of the transaction |  |
| Transaction\_time | Time | Time in HH:MM:SS |  |
| Transaction\_qty | Integer | How many products bought |  |
| Store\_id | Integer | Store ID in which the transaction happened | Three ID : 3,5,8 |
| Store\_Location | String | Store location in which the transaction happened | Three Store location :  Hell kitchen, Lower Manhattan,Astoria |
| Product\_id | Integer | Product Id of the product | Unique Id for every product, bit confusing since it is based on the Sub Type |
| Unit\_price | Float | Price per product |  |
| Product\_category | String | Which category the product is |  |
| Product\_type | String | Product type |  |
| Product\_detail | String | Product sub\_type | This Refers to the actual product |
| revenue | Float | Total cost per transaction |  |
| Month | Integer | Month in Number | Repitition of data |
| Month2 | String | Month in string | Repitition of data.The Name Could be changed |
| Weekday | Integer | Weekday in number | Repitition of data |
| Weekday3 | String | Weekday in string | Repitition of data .The Name Could be changed |
| Hour | Integer | Weekday in number | Repitition of data |

**Advantages:**

* Compared to the previous data set this has Rigid Schema
* Easy to query
* Only one table is available in the data set. No need to use complex queries like JOIN
* Primary key availability makes easy to uniquely identify each data.
* Very Detailed Report
* No null values and no duplicates, error

**Disadvantages**

* Semi normalized data. There is no parent child relationship
* Difficult to make changes without impacting the entire database
* This could easily have different tables like product\_table, store\_table, which would have made it very easy to analyze.

**DATA SET 3 – The Paris 2024 Olympic**

*The Paris 2024 Olympic Summer Games dataset provides comprehensive information about the Summer Olympics held in 2024. It covers various aspects of the event, including participating countries, athletes, sports disciplines, medal standings, and key event details.*

* Analyze the total number of medals won by each country
* Identify the countries with the highest medal counts and compare their performance to previous Olympics.
* Examine which athletes contributed the most to their country’s medal tally.
* Compare the performance and participation rates of male and female athletes across different sports.
* Determine which sports or disciplines had the most events or participants.

**Source :** <https://www.kaggle.com/datasets/piterfm/paris-2024-olympic-summer-games/data>

A screenshot of a computer

Description automatically generated

**TABLE DETAILS: Medallists**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data\_Type** | **Description** | **Observation** |
| Medal\_Date | Date | Date of the event |  |
| Medal\_Type | String | Gold, silver, Bronze |  |
| Medal\_code | Integer | Code : 1 2 3 |  |
| Name | String | Name of the athlete | Could be First name and last name |
| Gender | String | Gender of the athlete |  |
| Country\_code | String | Unique Country Code – 3 letters |  |
| Country | String | Country name in short |  |
| Country\_long | String | Full Country name |  |
| Nationality | String | Their nationality |  |
| Team | String | Their team | No Data |
| Team\_gender | String | Team gender | No Data |
| Discipline | String | Category of sport |  |
| Event | String | Event Name |  |
| Event\_type | String | 6 types |  |
| Url\_event | String | Event Url and Source of data | Can be Deleted. |
| Birth\_date | Date | Date of birth of athletes |  |
| Athlete\_code | Integer | Code of every athletes | Primary key |

**Advantages:**

* Rigid Schema
* Easy to query
* Primary key availability makes easy to uniquely identify each data.
* Very Detailed Report
* No null values and no duplicates, error
* Parent child relationship available.
* Normalized data sets with different tables

**Disadvantages**

* The child tables are not consistent data. They are raw data.

A screenshot of a computer

Description automatically generated

**General Advantages and Disadvantages of Having a Schema**

**Advantages:**

1. Data Integrity and Consistency:
   * A schema enforces rules and constraints that ensure data integrity, such as data types, primary keys, and foreign keys. This prevents invalid data from being entered and maintains consistency across the database.
2. Efficient Data Management:
   * Schemas provide a clear structure for organizing data, making it easier to manage, query, and maintain. This structure supports optimized storage, faster data retrieval, and more efficient updates.
3. Scalability and Flexibility:
   * Schemas provide a framework that supports the addition of new tables, fields, or relationships without disrupting existing data. This flexibility is crucial for scaling databases as organizational needs grow.

**Disadvantages:**

1. Rigidity:
   * Once a schema is defined, it can be difficult to make changes without impacting the entire database. Altering a schema often requires significant effort and can lead to downtime or data migration challenges.
2. Initial Complexity and Development Time:
   * Designing a comprehensive schema can be complex and time-consuming. It requires careful planning and a deep understanding of the data, which can delay the implementation of the database.
3. Maintenance Challenges:
   * As data needs evolve, maintaining and updating the schema can become challenging. Any changes to the schema may require careful reorganization of existing data, leading to potential disruptions.

**Impact on Data Processing and Analysis**

**Positive Impact:**

1. **Optimized Query Performance**:
   * A well-designed schema enables efficient data processing by organizing data in a way that facilitates faster querying. Indexes and relationships defined by the schema allow for quick access to relevant data, reducing the time needed for data retrieval.
2. **Enhanced Data Accuracy**:
   * Schemas enforce data integrity, which leads to more accurate and reliable data. This accuracy is crucial for data analysis, as it ensures that the insights drawn from the data are based on valid and consistent information.
3. **Improved Data Analysis Consistency**:
   * Consistent data types and relationships defined by the schema allow analysts to perform complex analyses with confidence. The standardized structure ensures that data can be aggregated, filtered, and compared across different dimensions without inconsistencies.

**Negative Impact:**

1. **Reduced Flexibility in Data Processing**:
   * The rigidity of a schema can limit the ability to adapt to new data processing requirements. If the schema does not accommodate certain types of data or new analytical needs, it may hinder the ability to perform certain analyses.
2. **Potential for Slow Performance**:
   * In cases where the schema is overly complex or where constraints are too strict, data processing can become slow, particularly for large datasets.

**Conclusion :**

In summary, structured data types and well-defined schemas are essential components of effective data management, ensuring that data is stored, processed, and analyzed in a reliable, efficient, and scalable manner.

while having a schema offers numerous advantages in terms of data integrity, management, and analysis, it also introduces challenges related to flexibility, performance, and maintenance. The impact on data processing and analysis is largely positive, but the potential drawbacks must be managed carefully to ensure that the schema continues to support the organization’s evolving data needs.