Data Modelling – YouTube Analytics

# Star & Snowflake Schema Design

The industry best practice for designing analytics data warehouses is to use star or snowflake schema design.

## Star Database Schema Design:

The star database schema design is a type of database modeling where data is organized around a central fact table connected to multiple dimension tables. The fact table contains numerical or quantitative data (facts), and the dimension tables store descriptive data about the facts. The relationship between the fact table and dimension tables forms a star-like structure, with the fact table at the center and dimension tables radiating outwards.

Benefits of Star Database Schema Design:

* Simplified Querying: Star schema simplifies querying as it avoids complex joins between tables, making queries more efficient and faster.
* Improved Performance: Star schema's denormalized structure enables faster data retrieval, especially for analytical queries involving aggregations.
* Data Aggregation: Aggregating data across dimensions is straightforward, allowing for quick generation of reports and analysis.
* Enhanced Scalability: Star schema's simplicity and efficient querying make it scalable and well-suited for handling large datasets.
* Easy Understandability: The star schema's intuitive design makes it easier for users to understand and work with the data.
* Better Analytical Capabilities: The schema facilitates multidimensional analysis and OLAP (Online Analytical Processing) applications.
* Data Warehousing: Star schema is commonly used in data warehousing environments, providing an efficient structure for analytical workloads.

## Snowflake Database Schema Design:

The snowflake database schema design is an extension of the star schema, where dimension tables are further normalized into sub-dimension tables. The normalized structure results in a snowflake-like appearance, with dimension tables branching out into multiple levels of related sub-dimension tables.

Benefits of Snowflake Database Schema Design:

* Reduced Redundancy: Snowflake schema minimizes data redundancy by normalizing dimension data, leading to reduced storage requirements.
* Data Consistency: Normalization ensures data consistency, avoiding data anomalies and improving data accuracy.
* Easier Maintenance: The snowflake schema's normalized structure makes data maintenance and updates more manageable.
* Improved Update Performance: Snowflake schema's normalized design often leads to faster updates and inserts compared to star schema.
* Enforced Referential Integrity: The normalized structure allows for better referential integrity enforcement between related tables.
* Query Optimization: Although more complex joins are required in snowflake schema compared to star schema, modern database engines can optimize query performance effectively.
* Data Warehouse Management: Snowflake schema is suitable for data warehouses with large datasets and complex relationships.

## Choosing Between Star and Snowflake Schema:

The decision between using a star or snowflake schema depends on the specific requirements and characteristics of the data and the analytical workload. Star schema is simpler and generally more suitable for data warehousing and OLAP environments, where performance and ease of querying are paramount. On the other hand, snowflake schema is preferred when data normalization is essential to reduce redundancy and improve data consistency.

# Identifying the data entities and relationships

Before we start with database design and data modelling we need to identify some data entities for the YouTube data and the relationships between them:

## Data entities list

* **Customer/User**: This entity represents a user of YouTube. It includes information such as the user's name, email address, and date of birth.
* **Video**: This entity represents a video that is hosted on YouTube. It includes information such as the video's title, description, and view count.
* **Channel details**: This entity represents a collection of videos that are owned by a single user. It includes information such as the channel's name, description, and subscriber count.
* **Comment**: This entity represents a comment that is made on a YouTube video. It includes information such as the comment's text, the commenter's name, and the comment's date.
* **Like**: This entity represents a like that is given to a YouTube video. It includes information such as the user who liked the video and the date the like was given.
* **Dislike**: This entity represents a dislike that is given to a YouTube video. It includes information such as the user who disliked the video and the date the dislike was given.
* **Playlist**: This entity represents a collection of videos that are organized together. It includes information such as the playlist's name, description, and number of videos.
* **Trending**: This entity represents a list of videos that are currently trending on YouTube. It includes information such as the videos' titles, view counts, and like/dislike ratios.

## Relationships between the data entities

* **Customer and Video**: A customer can have many videos, and a video can be owned by only one customer. This is a one-to-many relationship.
* **Customer and Comment**: A customer can comment on many videos, and a video can have many comments. This is a many-to-many relationship.
* **Video and Channel details**: A video belongs to one channel, and a channel can have many videos. This is a many-to-one relationship.
* **Video and Like**: A user can like many videos, and a video can be liked by many users. This is a many-to-many relationship.
* **Video and Dislike**: A user can dislike many videos, and a video can be disliked by many users. This is a many-to-many relationship.
* **Video and Playlist**: A video can be included in many playlists, and a playlist can contain many videos. This is a many-to-many relationship.
* **Video and Trending**: A video can be trending, and a trending video can be one of many videos. This is a many-to-one relationship.

# Conceptual Data Model

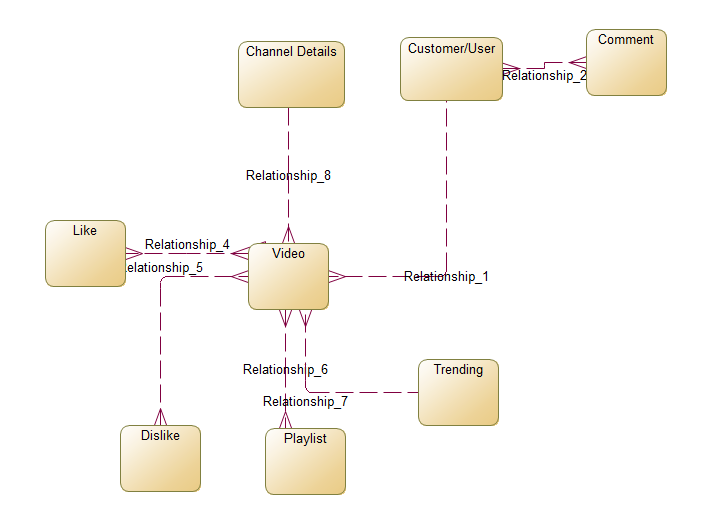
A conceptual data model is an abstract representation of an organization's data requirements, independent of any specific database management system or technology. It provides a high-level view of the essential data entities, their attributes, and the relationships between them. The primary focus of a conceptual data model is on understanding the business concepts and data semantics, enabling effective communication between business stakeholders and data professionals.

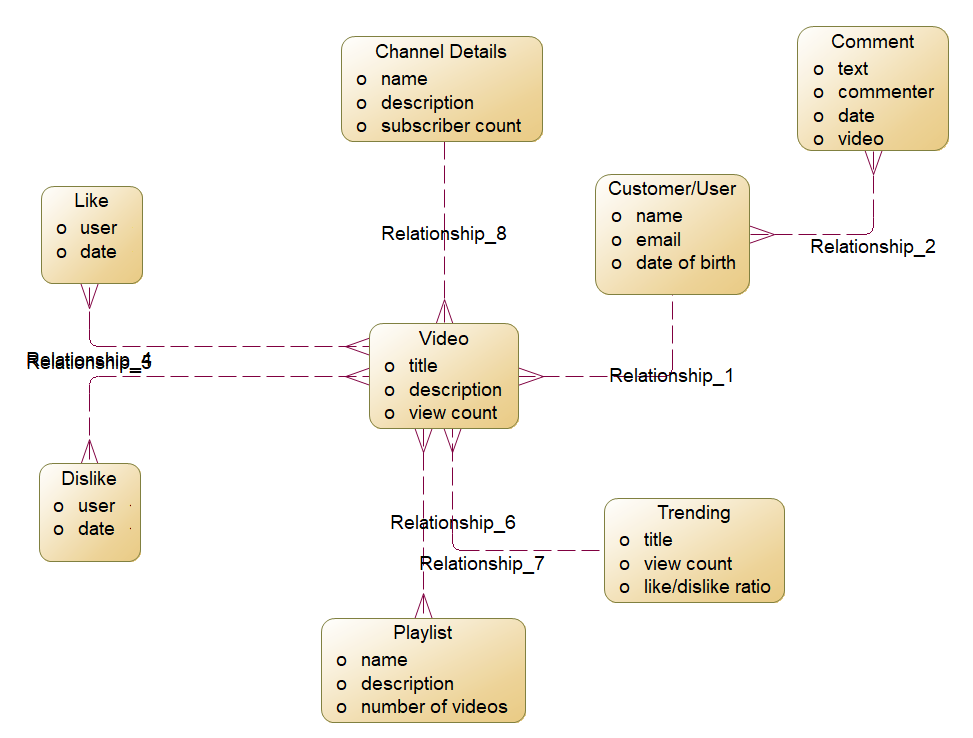
Benefits of Conceptual Data Model:

* **Clarity in Data Understanding**: A conceptual data model offers a clear and concise representation of the organization's data requirements. It helps stakeholders to understand the fundamental data concepts, relationships, and business rules without getting into technical details.
* **Improved Communication**: The model serves as a communication bridge between business users and technical teams. It allows business stakeholders to express their data needs in business terms, while technical teams can map those requirements to the underlying data structures.
* **Business Alignment**: By focusing on the business aspects of data, the conceptual data model ensures that data management efforts are aligned with the organization's strategic objectives and operational needs.
* **Reduced Ambiguity**: The model defines data entities and their relationships, reducing ambiguity and misunderstandings about data definitions and semantics.
* **Efficient Data Integration**: A well-designed conceptual data model aids in understanding data integration requirements between different systems, facilitating data sharing and interoperability.
* **Basis for Logical and Physical Data Models**: The conceptual data model serves as a foundation for creating logical and physical data models, which map the conceptual representation into actual database structures.
* **Data Governance and Compliance**: A conceptual data model provides a consistent and standardized approach to data representation, contributing to better data governance and regulatory compliance.
* **Scalability and Flexibility**: As an abstract representation, the conceptual data model allows for changes and modifications to be made more easily. It offers the flexibility to adapt to changing business needs and accommodate future data requirements.
* **Requirements Validation**: The conceptual data model serves as a basis for validating data requirements against business processes. It helps identify data gaps and redundancies early in the project lifecycle.
* **Simplification of Data Architecture**: By focusing on the essential business entities and relationships, the conceptual data model simplifies data architecture design and reduces unnecessary complexity.
* **Consistency across Projects**: A well-defined and standardized conceptual data model promotes consistency in data representation across various projects and initiatives within the organization.
* **Data Quality Improvement**: The model lays the groundwork for data quality management efforts by identifying data entities that require higher data quality standards and attention.

The conceptual data model plays a vital role in bridging the gap between business understanding and technical implementation in data management initiatives. Its benefits encompass improved communication, enhanced data quality, and the establishment of a solid foundation for subsequent stages of data modeling and system development.

Conceptual data model using the data entities for YouTube analytics.





# Logical Data Model

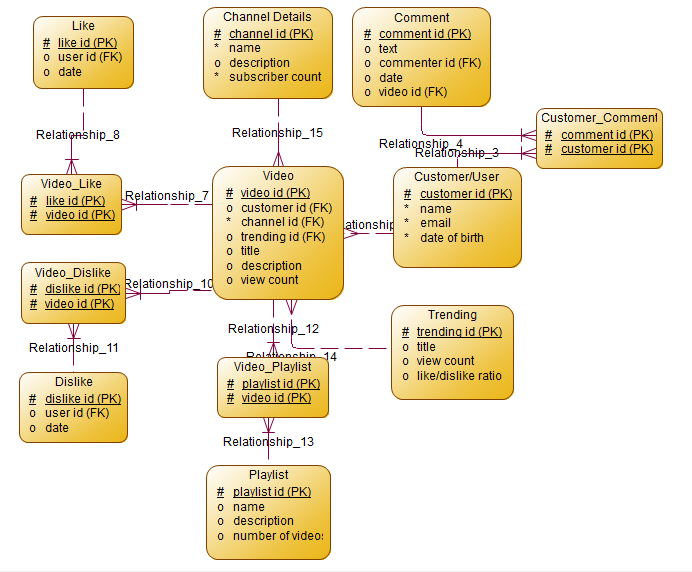
A logical data model is a detailed and technology-independent representation of an organization's data requirements. It defines the logical structure of data entities, their attributes, relationships, and business rules. Unlike the conceptual data model, the logical data model takes into account the semantics and interrelationships of data elements, providing a blueprint for database design and implementation. It serves as an intermediate step between the conceptual data model and the physical data model, facilitating the translation of business requirements into technical specifications.

Benefits of Logical Data Model:

* Data Standardization: The logical data model establishes standardized data definitions, ensuring consistency and uniformity in data representation across different systems and applications.
* Business Rule Clarity: By capturing detailed business rules, the logical data model ensures that data is handled and processed in accordance with specific business requirements and constraints.
* Elimination of Data Redundancy: The model identifies potential data redundancy, helping to eliminate duplicate data and optimize storage resources.
* Efficient Data Integration: A well-designed logical data model simplifies data integration efforts, as it provides a comprehensive understanding of data relationships and dependencies.
* Improved Data Quality: By specifying data constraints and validation rules, the logical data model contributes to better data quality, reducing errors and inconsistencies in data entry and processing.
* Database Design Guidance: The logical data model serves as a guide for database designers and developers, facilitating the design and creation of an efficient database structure.
* Data Integrity Assurance: The model defines data integrity rules and constraints, ensuring that data remains accurate and consistent throughout its lifecycle.
* Optimized Performance: The logical data model helps optimize database performance by identifying potential bottlenecks and designing efficient data access paths.
* Data Security Considerations: By identifying sensitive data elements and their relationships, the logical data model aids in designing appropriate data security measures.
* Data Governance Alignment: The logical data model aligns data governance efforts with specific data entities and attributes, enabling better data management and oversight.
* Impact Analysis: The model allows for effective impact analysis when changes are proposed, helping assess the effects of modifications on other data elements and processes.
* Collaboration with Stakeholders: The logical data model provides a platform for collaboration between business stakeholders, data analysts, and IT professionals, fostering a shared understanding of data requirements.
* Data Migration Planning: The logical data model assists in planning data migration and conversion processes, ensuring seamless transitions between systems.
* Documentation and Knowledge Transfer: The model serves as a valuable documentation resource for future reference and aids in knowledge transfer between team members.
* Agile Development Support: The logical data model allows for iterative development, making it easier to adapt to changing business needs and incorporate user feedback.

The logical data model plays a pivotal role in translating business requirements into detailed technical specifications for database design and implementation. Its benefits span from improved data quality and performance to streamlined data integration and governance. By establishing a clear and comprehensive logical data model, organizations can create robust, efficient, and well-structured databases that align with their business objectives and support data-driven decision-making.

The logical data model derived from the conceptual data model for the YouTube analytics exercise.



# Physical Data Model

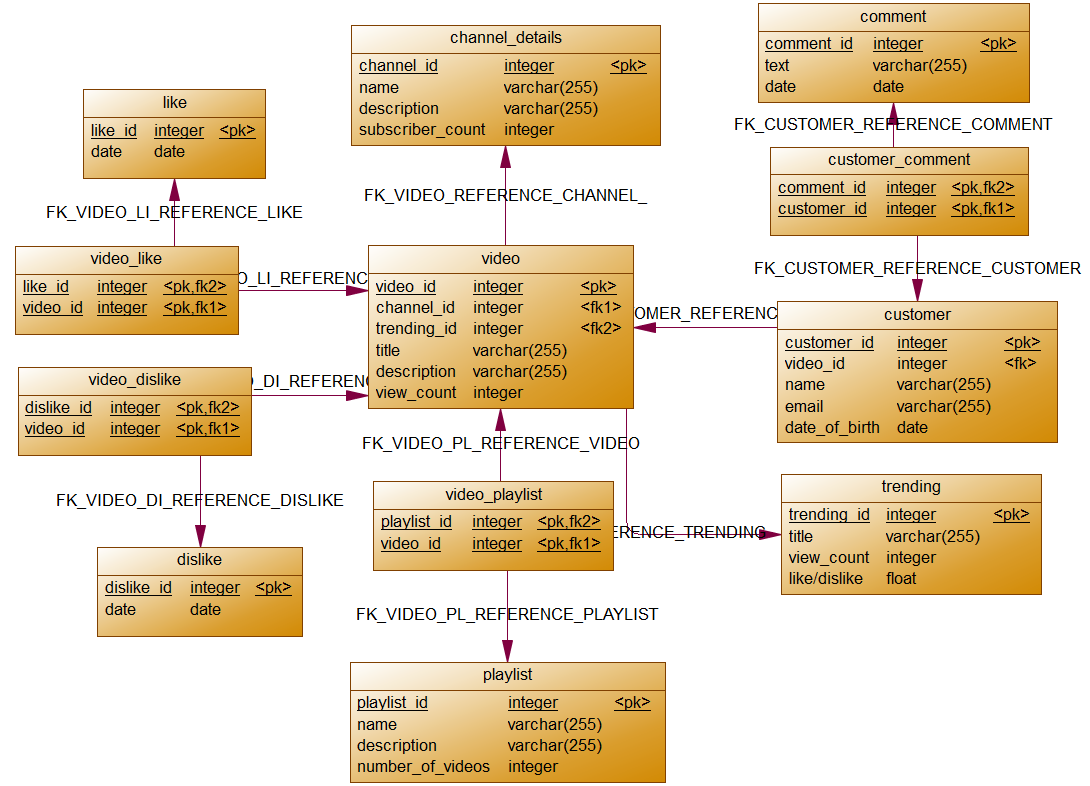
A physical data model is a technical representation of an organization's data requirements, detailing how the data will be stored, organized, and implemented in a specific database management system. It involves transforming the logical data model into a database schema, specifying data types, indexes, constraints, and storage details. The physical data model is closely tied to the chosen database technology and addresses performance, scalability, and data storage considerations.

Benefits of Physical Data Model:

* Efficient Database Design: The physical data model optimizes database design for a particular database management system, taking advantage of its specific features and capabilities.
* Performance Optimization: By defining appropriate indexes, partitioning, and data structures, the physical data model enhances database performance, resulting in faster data retrieval and processing.
* Data Storage Efficiency: The model ensures efficient data storage by specifying the appropriate data types and storage requirements, minimizing storage space usage.
* Normalization and Denormalization: The physical data model strikes the right balance between normalization (reducing redundancy) and denormalization (improving performance) to support efficient data retrieval while maintaining data integrity.
* Data Integrity Enforcement: The model enforces data integrity constraints, such as primary keys, foreign keys, and check constraints, to maintain data consistency and accuracy.
* Database Security Measures: The physical data model incorporates security considerations, specifying access controls, authentication mechanisms, and encryption methods to protect sensitive data.
* Data Backup and Recovery: The model defines data backup and recovery strategies, ensuring data availability and reliability in the event of system failures or data loss.
* Data Partitioning and Distribution: The physical data model determines data partitioning and distribution strategies, optimizing data management for large-scale and distributed systems.
* Scalability Support: By considering scalability requirements, the model allows for seamless expansion as data volume and user demands grow over time.
* Database Maintenance Planning: The physical data model aids in database maintenance planning, guiding tasks like index rebuilding, statistics updates, and data archiving.
* Data Migration and Implementation Guidance: The model provides clear guidelines for data migration from development to production environments, ensuring a smooth implementation process.
* Collaboration between IT Teams: The physical data model fosters collaboration between database administrators, developers, and data engineers, enabling effective communication and coordination.
* Regulatory Compliance: The model supports regulatory compliance efforts by implementing data storage and security measures that align with relevant industry standards and regulations.
* Disaster Recovery Planning: The physical data model contributes to disaster recovery planning, enabling the organization to restore data and services in case of catastrophic events.
* Documentation and Maintenance: The physical data model serves as a crucial documentation resource, facilitating ongoing maintenance, troubleshooting, and system enhancements.

The physical data model plays a pivotal role in transforming the logical data model into a well-optimized, secure, and efficient database implementation. Its benefits range from improved performance and data storage efficiency to supporting data integrity and regulatory compliance. By following the guidelines provided by the physical data model, organizations can build robust and scalable database systems that meet their specific technical requirements while aligning with their business objectives.

Physical data model derived from the logical data model for the YouTube analytics exercise.



Database Diagram in SQL Server

