A Unified Relational Database Management System for Heterogeneous Computer Vision Datasets

Ayush Vaibhav Bhatti[1], Mahshad Akbarsharifi[2], and Ariel Athena Moncrief [3]

[1]Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ, USA
{ayushbhatti, mahshadsharifi, arielmoncrief}@arizona.edu

December 15, 2025

Abstract

Modern computer vision workflows are heavily based on large annotated datasets such as MS COCO, PASCAL VOC, and OpenImages. However, these datasets differ substantially in structure, annotation format, semantics, and scale, making unified analysis and efficient querying difficult. In this project, we design and implement a unified relational database management system that integrates three major computer vision datasets into a single normalized schema. We develop data set-specific ETL pipelines to convert JSON, XML, and CSV annotations into a standardized SQL representation, enabling consistent storage of images, categories, bounding-box annotations, and metadata. We also implement a suite of analytical SQL queries to examine the characteristics of the data set, detect annotation errors, and quantify distributional properties such as class imbalance, object-size variability, and annotation density. To evaluate the system, we compare query performance with and without indexing, visualize dataset statistics through plots, and analyze insights derived from unified querying across datasets. The resulting system demonstrates that traditional relational databases can effectively support the integration of large-scale CV datasets, improve data quality diagnostics, and enable cross-dataset analytics without requiring additional preprocessing. Our framework provides a foundation for downstream training pipelines and future extensions in data management for machine learning.

1 Introduction

Large-scale annotated datasets form the backbone of modern computer vision research and practice. Benchmarks such as MS COCO, PASCAL VOC, and OpenImages have

enabled rapid progress in object detection, instance recognition, and scene understanding. However, despite their widespread adoption, these datasets are highly heterogeneous, differing in annotation format (JSON, XML, CSV), schema structure, category taxonomies, metadata conventions, coordinate systems, and overall scale. As a result, integrating or jointly analyzing multiple datasets remains a non-trivial challenge. Researchers and practitioners frequently build ad-hoc scripts to parse each dataset independently, making data exploration, debugging, and cross-dataset analytics slow, error-prone, and difficult to reproduce.

This project addresses the core question: *Can we unify heterogeneous computer vision datasets into a single relational database that supports efficient querying, analysis, and data-quality diagnostics?*

To answer this, we design a normalized relational schema capable of representing images, categories, bounding-box annotations, dataset metadata, and segmentation information across COCO, VOC, and OpenImages. We develop dataset-specific ETL pipelines to convert raw annotations into a standardized SQL representation, ensuring consistency while preserving dataset-specific semantics. Building on this unified representation, we implement a rich set of analytical SQL queries to measure class imbalance, bounding-box distributions, annotation density, segmentation coverage, and dataset scale. We further investigate query performance, evaluating the effect of indexing strategies on large annotation tables. Our results show that classical relational database systems can effectively support machine-learning-centric data management tasks. A unified schema not only simplifies cross-dataset queries but also reveals meaningful dataset properties, detects annotation errors, and provides insights relevant for training pipelines and dataset curation. This work demonstrates that database principles—normalization, indexing, error checking, and declarative querying—remain highly valuable tools in modern computer vision workflows.

2 Task Definition

The goal of this project is to design and implement a unified database management system capable of storing, querying, and analyzing multiple heterogeneous computer vision datasets. Specifically, we focus on integrating three widely used benchmarks—MS COCO, PASCAL VOC 2007, and OpenImages V7—which differ substantially in annotation format, schema design, and scale. Each dataset provides object detection annotations but uses incompatible representations: COCO adopts a JSON-based hierarchical structure, VOC relies on XML files, and OpenImages distributes annotations across multiple CSV tables. These differences make direct comparison, joint analysis, and large-scale data exploration cumbersome without substantial preprocessing. Formally, the task can be defined as follows: given raw dataset files D consisting of JSON, XML, and CSV annotation formats, the system must produce a unified relational database U that supports declarative SQL queries over images, categories, and bounding-box annotations. The expected system behavior can be described as:

- **Input:** Raw annotation files from COCO (JSON), VOC (XML), and OpenImages (CSV), along with corresponding metadata such as categories, splits, and image dimensions.

- **Output:** A normalized SQLite database containing tables for datasets, images, categories, annotations, and segmentation placeholders, with foreign-key constraints enforcing relational integrity.
- **Functionality:** Support expressive SQL queries for dataset statistics, cross-dataset comparisons, annotation validation, and large-scale analytical tasks.

To evaluate the system, we define three criteria. First, **correctness** requires successful ingestion of all available annotation records into the unified schema without loss of information. Second, **expressiveness** requires that the schema supports typical analysis tasks such as class-frequency computation, annotation-density measurement, bounding-box distribution analysis, and data-quality diagnostics. Third, **performance** evaluates whether indexing strategies improve query latency on large annotation tables. The scale of the datasets—over 120,000 COCO images, 5,000 VOC images, and 17,000 OpenImages samples—provides a realistic setting for analyzing the challenges of data unification, modeling, and optimization in database systems.

3 Related Work

The growing scale and diversity of computer vision datasets have motivated research across annotation efficiency, structured data representation, and machine-learning-driven data analysis. Existing work highlights the importance of reducing annotation overhead, improving data organization, and enabling scalable analytics, but often lacks a unified data management abstraction across heterogeneous datasets.

3.1 Annotation Efficiency and Resource-Aware Dataset Design

Several studies focus on reducing annotation cost and memory usage in large-scale or resource-constrained vision systems. Memory-efficient annotation strategies have been explored in domains such as autonomous drone navigation, where storage limitations necessitate compact representations and selective labeling of visual data [2]. These approaches demonstrate that careful annotation design can significantly reduce memory overhead while preserving task performance. However, such methods typically operate at the application or algorithmic level and do not provide a general-purpose framework for managing annotations across multiple datasets with differing schemas and formats. Related work in biomedical imaging similarly emphasizes structured annotation and data integrity to support large-scale analysis and reproducibility [3]. These systems show the effectiveness of normalized schemas and metadata validation but are largely domain-specific and do not address the heterogeneity of general-purpose computer vision benchmarks.

3.2 Machine Learning–Driven Analysis of Visual and CV Data

Another line of research applies machine learning techniques to extract structured information from unstructured documents such as curriculum vitae and visual records.

Recent work demonstrates that computer vision and learning-based pipelines can successfully transform CV data into structured representations suitable for predictive modeling, including personality and trait inference [1]. While these studies highlight the feasibility of automated data extraction and analysis, they primarily emphasize model performance and prediction accuracy. Data ingestion, validation, and organization are handled through task-specific preprocessing pipelines rather than reusable, database-backed infrastructures. More broadly, data-centric AI research increasingly recognizes that dataset quality, annotation consistency, and distributional properties strongly influence downstream model performance. Despite this, most existing systems rely on ad-hoc scripts and flat-file storage, limiting reproducibility, cross-dataset comparison, and systematic data diagnostics.

3.3 Database-Oriented Data Management for Vision Pipelines

Traditional relational database techniques—such as normalization, indexing, and declarative querying—have been widely adopted in classical data management but remain underutilized in computer vision workflows. Prior approaches demonstrate that structured storage and integrity constraints can simplify analytics and error detection, yet few systems explicitly target the unification of heterogeneous CV datasets spanning multiple annotation formats (JSON, XML, CSV), coordinate systems, and category taxonomies.

3.4 Positioning of Our Work

Our work bridges these gaps by introducing a unified relational database management system that integrates multiple large-scale computer vision datasets within a single normalized SQL schema. Unlike prior annotation- or model-centric approaches, our framework emphasizes dataset-agnostic storage, integrity enforcement, and query-driven analytics. This enables systematic exploration of class imbalance, annotation density, and data quality across datasets, providing a scalable foundation for data-centric machine learning and downstream training pipelines.

4 Dataset Description

This project integrates three widely used computer vision datasets—MS COCO 2017, PASCAL VOC 2007, and OpenImages V7—each of which provides object detection annotations but differs substantially in scale, annotation structure, taxonomy, and file format.

MS COCO 2017. COCO contains 118,287 training images and 5,000 validation images, annotated with over 896,000 object instances across 80 categories. Annotations are stored in nested JSON files that encode images, categories, bounding boxes, and segmentation masks. COCO follows a consistent pixel-based coordinate system and

provides rich metadata such as `iscrowd` flags, segmentation polygons, and supercategories. Its large scale and annotation richness make COCO a challenging but representative benchmark for unified data management.

PASCAL VOC 2007. VOC 2007 consists of 9,963 images labeled across 20 object categories. Unlike COCO, VOC uses individual XML files for each image, where annotations follow a flat hierarchical structure. Bounding boxes are defined using integer pixel coordinates, and metadata is minimal compared to COCO. VOC's relatively small size and simpler schema highlight the difficulties of harmonizing datasets with incompatible annotation formats.

OpenImages V7 (Subset). OpenImages is a large-scale dataset with millions of images; due to computational constraints, we import a curated subset of 17,125 images along with 144,803 bounding-box annotations. OpenImages distributes its annotation data across multiple CSV files, using normalized bounding-box coordinates in the range $[0, 1]$. Category labels are defined by unique machine-generated identifiers (MIDs) rather than human-readable names, requiring additional mapping. The dataset's distributed CSV structure and large category taxonomy introduce unique challenges for schema normalization and ETL processing.

Heterogeneity Challenges. Together, these datasets cover three different annotation formats (JSON, XML, CSV), two coordinate conventions (absolute pixels vs. normalized), and three different category semantics (COCO-style categories, VOC categories, and OpenImages MID taxonomy). This heterogeneity motivates the need for a unified relational schema capable of reconciling differences across annotation structure, metadata, and scale. Table 1 Summary statistics for datasets used in this project.

5 System Infrastructure

The goal of the system infrastructure is to unify heterogeneous computer vision datasets into a single, normalized SQL database while enabling efficient querying and scalable analytics. Our pipeline consists of three main components: (1) dataset-specific ETL processing, (2) construction of a unified relational schema, and (3) SQL-based analytics for downstream evaluation. Figure 1 provides an overview of the full architecture.

5.1 End-to-End Architecture

Each dataset (COCO, VOC, OpenImages) is ingested through a custom Python ETL script that parses its native annotation format and transforms the data into a standardized structure. COCO annotations are read from JSON, VOC annotations from XML, and OpenImages annotations from multiple CSV files. These scripts normalize category identifiers, bounding-box conventions, and metadata fields to ensure consistency across datasets. The processed outputs are then inserted into a unified SQLite database that enforces relational integrity through foreign keys. Once the database is

constructed, SQL queries are used to perform distributional analysis, detect annotation errors, compute dataset statistics, and compare datasets. This separation between data ingestion and analytical querying ensures modularity, reproducibility, and extensibility. Additional datasets may be incorporated simply by writing a compatible ETL script and populating the shared relational schema.

5.2 Relational Schema Design

The unified relational schema (Figure 2) is designed around five core tables: `Dataset`, `Image`, `Category`, `Annotation`, and `Segmentation`. The schema follows third-normal-form principles to eliminate redundancy while preserving efficient join operations for analytical workloads.

Dataset Table. Stores high-level metadata such as dataset name, version, and description. Every image and category is associated with exactly one dataset via a foreign key.

Image Table. Contains an internal image identifier, dataset identifier, file path or URL, image dimensions (when available), and the dataset split (train/val/test). Each row corresponds to a unique image across any dataset.

Category Table. Represents object categories with dataset-specific label identifiers. The table supports both human-readable category names (e.g., COCO labels) and machine-level external identifiers (e.g., OpenImages MIDs).

Annotation Table. Stores all bounding-box annotations using a dataset-agnostic representation: $(x_{\min}, y_{\min}, \text{width}, \text{height})$. Additional fields include the bounding-box area, difficulty flags, is-crowd indicators, and dataset-specific metadata. Each annotation links to a corresponding image and category.

Segmentation Table. Although not used for all datasets, this table supports COCO-style segmentation masks, stored either as polygon lists or compressed encodings when available.

This schema supports efficient cross-dataset queries, such as comparing category frequencies, computing bounding-box distributions, and identifying annotation inconsistencies.

5.3 Indexing and Performance Considerations

To improve query performance on large annotation tables—especially for COCO—we introduce indexes on foreign-key fields and frequently queried attributes. Indexes on `Annotation(image_id)`, `Annotation(category_id)`, and `Image(dataset_id, split)` significantly reduce join latencies and enable scalable analytics. This design choice is essential for supporting multi-dataset queries and ensuring that exploratory analysis remains responsive even on large-scale datasets.

6 Query Design

A central goal of the unified database is to support expressive, dataset-agnostic analysis through declarative SQL queries. The design of our query suite therefore focuses on extracting meaningful statistical, structural, and diagnostic information from large-scale annotation tables. By constructing queries over a normalized schema, we enable cross-dataset comparisons, data-quality auditing, and exploratory analysis without additional preprocessing or dataset-specific logic. The analytical queries implemented in this work fall into three major categories:

1. **Distributional structure of object categories**, including measures of class imbalance, category frequency, and annotation density across dataset splits.
2. **Data completeness and annotation coverage**, such as detecting unlabeled images, missing categories, and gaps in annotation consistency.
3. **Data quality diagnostics**, which identify invalid bounding boxes, malformed annotations, or other structural inconsistencies within the datasets.

These queries are essential for understanding dataset bias, ensuring annotation integrity, and preparing the unified dataset for downstream machine-learning pipelines. Representative examples are provided below.

6.1 Category Frequency and Imbalance

Listing 1 shows the SQL query used to compute the most frequent categories by number of images in which they appear. This query is used to generate the Top-10 plot in Section 8. In addition to the Top-10 analysis, Listing 2 presents a core query that computes the number of unique images per category across all datasets. This enables examination of category frequency at a finer granularity while preserving dataset provenance. This query exposes the long-tailed distribution characteristic of large-scale computer vision datasets, where a small subset of categories accounts for a disproportionate fraction of annotated images. Such imbalance has a direct impact on model generalization, often biasing learning toward high-frequency classes. In our implementation, categories originating from the OpenImages dataset dominate the highest-frequency classes, while the PASCAL VOC dataset contributes several low-frequency (rare) categories. This cross-dataset imbalance further amplifies class skew and motivates the need for imbalance-aware modeling and evaluation strategies. This behavior is further illustrated in Figure 3, which compares the Top-10 categories by image frequency across COCO, OpenImagesV7, and PASCAL VOC 2007.

6.2 Bounding-Box Area by Category

To examine the scale of objects across categories, we compute the average bounding-box area per category restricted to a particular dataset (e.g., COCO). Listing 3 highlights differences in object scale (e.g., large objects such as trains or buses versus small objects such as bottles or bowls).

6.3 Annotation Density per Split

We also design queries to measure annotation density across dataset splits. Listing 4 computes the average number of annotations per image for each split. This result is later visualized in Figure 6.

6.4 Data Completeness and Error Diagnostics

Two additional query patterns focus on data completeness and annotation integrity. Listing 5 identifies images with no associated annotations. Listing 6 detects invalid bounding boxes (e.g., negative dimensions or zero-area annotations). These diagnostic queries directly support data-quality analysis presented in Section 8.

7 Query Optimization

While the unified dataset enables expressive analysis, naive execution of analytical queries scales poorly due to the size of annotation tables, particularly for COCO and OpenImages. To support interactive exploration, we introduce several index structures targeting high-frequency join and group-by operations.

7.1 Indexing Strategy

Most analytical queries rely on joins of the form:

$$\texttt{Annotation} \bowtie \texttt{Image} \bowtie \texttt{Category},$$

which necessitates fast lookup on foreign-key attributes. We therefore add indexes on:

- `Annotation(image_id)` — accelerates annotation-to-image joins;
- `Annotation(category_id)` — accelerates category-frequency queries;
- `Image(dataset_id, split)` — accelerates dataset-level and split-level aggregations.

The corresponding SQL statements are shown in Listing 7 These indexes are particularly important because the `Annotation` table contains over 900k rows for COCO alone and a substantial number of rows for the OpenImages subset.

7.2 Performance Impact

To quantify the effect of indexing, we compare the execution time of representative queries before and after index creation. For example, the category-frequency aggregation of Listing 1 executes approximately $8\times$ faster after indexing `Annotation (category_id)`. Similarly, computing annotation density across splits (Listing 4) improves by roughly $6\times$, and dataset-summary queries improve by about $4\times$. These results demonstrate that even traditional B-tree indexes substantially improve analytical performance under large-scale computer vision workloads.

7.3 Query Planning Considerations

SQLite's query planner benefits significantly from the reduced search space provided by selective indexes. Without indexing, join operations trigger full table scans, yielding near-quadratic behavior for multiway joins. With indexing, lookups become logarithmic in the number of rows, enabling near-interactive analysis even on large datasets. This reinforces the value of relational optimization techniques when managing machine-learning datasets.

8 Experiments and Analysis

We evaluate the unified database through a series of analytical queries and visualizations designed to measure dataset structure, annotation quality, and metadata completeness. Each experiment highlights a specific statistical property or diagnostic capability enabled by the relational schema.

8.1 Category Distribution and Class Imbalance

Figure 4 shows the frequency of the ten most common object categories in COCO, computed using Listing 1. As expected, the *person* category appears in a disproportionately large number of images, far exceeding any other class. This long-tailed structure is characteristic of real-world datasets and underscores the importance of designing sampling and augmentation strategies for downstream machine-learning tasks.

8.2 Bounding-Box Scale Distribution

To understand annotation geometry, we examine the distribution of bounding-box areas in COCO, using the query in Listing 3 and plotting per-instance areas. Figure 5 reveals a heavy skew toward small objects, with a long tail of large-area annotations. Such variation directly impacts detector design, anchor sizes, and feature-map resolution requirements.

8.3 Annotation Density Across Splits

Using the unified schema, we compute the average number of bounding boxes per image for each dataset split via Listing 4. Figure 6 shows that COCO's training split exhibits significantly higher annotation density than its validation split, suggesting stronger supervisory signals during training than during evaluation.

8.4 Dataset Scale Comparison

Figure 8 and Figure 7 show the proportional distribution of images and annotations across datasets, respectively. COCO dominates the unified corpus in both metrics, motivating the indexing strategies discussed in Section 6. VOC contributes a relatively small fraction of the total annotation volume, while the sampled OpenImages subset sits between the two in scale.

8.5 Annotation Integrity and Data Quality

We run diagnostic queries to detect malformed annotations and missing coverage. Listing 5 reveals images without bounding-box annotations, which may indicate either intentional negative examples or incomplete annotation passes. Listing 6 identifies zero-area bounding boxes, negative coordinates, and other invalid geometric configurations. Although such errors are relatively rare, their presence can degrade model performance and complicate evaluation. The unified schema allows these diagnostics to be executed using simple SQL conditions, demonstrating the practicality of relational abstractions for large-scale vision datasets.

8.6 Query Performance Scaling

Figure 9 illustrates the average execution time of three representative queries as the dataset size increases from 25% to 100%. Each query is executed across three configurations: a baseline schema where each dataset is stored in its own dedicated set of tables, a centralized schema without indexing, and a centralized schema with indexing. Execution times are recorded across 10 runs to reduce noise and obtain a better representative value.

Figure 9a demonstrates the execution times of Listing 1, which performs frequency-based aggregation to compute the top 10 categories by number of distinct images. As expected, execution time consistently increases with dataset size across configurations. The baseline and centralized non-indexed schemCas exhibit comparable performance at smaller dataset sizes (25%–50%), while the centralized indexed configuration performs slightly better. As size increases to 75% and 100%, the centralized indexed schema consistently achieves the lowest execution time, indicating that indexing can be beneficial for aggregation-heavy queries when aligned with indexed columns.

Figure 9b illustrates Listing 3 that involves large joins and numerical aggregation to compute the average bounding-box area per category. Compared to the baseline and non-indexed schemas, the indexing yields significantly higher execution times. At 100% of the dataset, the indexed configuration is x6-7 times slower than the other configurations. This increase in execution time is due to the need to process nearly all Annotation rows to compute average bounding-box areas. Querying indexed databases with large joins, SQLite utilizes indexed keys to perform index-based lookups to reference corresponding rows. Given an Annotation entry, SQL repeatedly fetches matching rows from joined tables via small, repetitive row accesses. These row accesses occur in index order rather than in sequential order, leading to poor locality and increased lookup overhead.

Figure 9c shows Listing 4, which incorporates a WITH clause and a left join to compute annotation density per dataset split. The centralized indexed schema consistently achieves the lowest execution time, with minimal increase as the size increases. The baseline schema, in contrast, shows the steepest growth in execution time between 50% and 100% of the dataset, becoming the most inefficient configuration at those respective dataset sizes. The improved performance of the indexed centralized configuration suggests that indexing is effective for grouping on image identifiers and for joining index-aligned predicates and join keys.

9 Discussion and Future Work

Our results demonstrate that a traditional relational database can effectively support the integration and analysis of heterogeneous computer vision datasets. The unified schema simplifies cross-dataset querying and enables systematic inspection of class distributions, annotation density, and data quality using concise SQL queries. Indexing plays a critical role in achieving interactive performance for aggregation-heavy workloads, particularly for large-scale datasets such as COCO and OpenImages. Despite these strengths, several limitations remain. First, SQLite is not designed for concurrent or large-scale multi-user workloads, limiting scalability beyond exploratory analysis. Second, while the schema includes a segmentation table, segmentation masks are not yet fully ingested or queried. Finally, categories are treated as dataset-specific entities, preventing direct semantic alignment across datasets. Future work includes migrating the system to a production-grade DBMS (e.g., PostgreSQL), extending support for segmentation annotations, introducing a shared category ontology across datasets, and exploring materialized views or alternative storage layouts to further optimize analytical performance.

10 Conclusion

We presented a unified relational database management system for integrating MS COCO, PASCAL VOC, and OpenImages by designing a normalized schema and dataset-specific Python ETL pipelines that convert heterogeneous JSON, XML, and CSV annotations into a consistent SQL representation, enabling analytical queries over category distributions, annotation density, dataset scale, and annotation quality, and demonstrating that conventional relational techniques such as indexing and declarative querying remain effective for preparing large-scale vision datasets for downstream machine-learning tasks and future data-centric AI research.

References

[1] Nandani Agarwal, MJ Akshaya, Nihar Shetty, Siddharth Kumar, et al. Machine learning driven personality prediction system using the concept of cv analysis. In *2024 International Conference on Electrical Electronics and Computing Technologies (ICEECT)*, volume 1, pages 1–6. IEEE, 2024.

[2] Pratibha Vittal Hegde, Mohammed Riyaz Ahmed, and Abdul Haq Nalband. Memory-efficient annotation techniques for autonomous drone navigation. *IEEE Access*, 2025.

[3] Adrian Krenzer, Kevin Makowski, Amar Hekalo, Daniel Fitting, Joel Troya, Wolfram G Zoller, Alexander Hann, and Frank Puppe. Fast machine learning annotation in the medical domain: a semi-automated video annotation tool for gastroenterologists. *BioMedical Engineering OnLine*, 21(1):33, 2022.

A Appendix

A.1 Tables

Table 1: Summary statistics for datasets used in this project.

Dataset	# Images	# Annotations	# Categories
COCO 2017 (train+val)	123,287	896,782	80
VOC 2007	5,011	15,662	20
OpenImages V7 (subset)	17,125	144,803	$\sim$600

A.2 Full SQL Listings

Listing 1: Top-10 categories by image frequency

```sql
SELECT c.name,
       COUNT(DISTINCT a.image_id) AS num_images
FROM Annotation a
JOIN Category c ON c.category_id = a.category_id
GROUP BY c.name
ORDER BY num_images DESC
LIMIT 10;
```

Listing 2: Category image frequency grouped by dataset

```sql
SELECT d.name AS dataset_name,
       c.name AS category_name,
       COUNT(DISTINCT a.image_id) AS image_count
FROM Annotation a
JOIN Category c ON c.category_id = a.category_id
JOIN Dataset d ON d.dataset_id = c.dataset_id
GROUP BY d.name, c.name
ORDER BY image_count DESC;
```

Listing 3: Average bounding-box area per category (COCO)

```sql
SELECT c.name,
       AVG(a.area) AS avg_area
FROM Annotation a
JOIN Image i    ON i.image_id = a.image_id
JOIN Category c ON c.category_id = a.category_id
JOIN Dataset d  ON d.dataset_id = i.dataset_id
WHERE d.name = 'COCO'
  AND a.area > 0
GROUP BY c.name
ORDER BY avg_area DESC;
```

Listing 4: Annotation density per split

```sql
WITH box_counts AS (
```

```
    SELECT image_id, COUNT(*) AS num_boxes
    FROM Annotation
    GROUP BY image_id
)
SELECT i.split,
       AVG(COALESCE(bc.num_boxes, 0)) AS avg_boxes_per_image
FROM Image i
LEFT JOIN box_counts bc ON i.image_id = bc.image_id
GROUP BY i.split
ORDER BY i.split;
```

Listing 5: Images without annotations

```
SELECT d.name AS dataset,
       i.split,
       COUNT(*) AS empty_images
FROM Image i
JOIN Dataset d ON d.dataset_id = i.dataset_id
LEFT JOIN Annotation a ON a.image_id = i.image_id
WHERE a.annotation_id IS NULL
GROUP BY d.name, i.split;
```

Listing 6: Detection of invalid bounding boxes

```
SELECT annotation_id, image_id,
       bbox_xmin, bbox_ymin,
       bbox_width, bbox_height
FROM Annotation
WHERE bbox_width <= 0 OR bbox_height <= 0
   OR bbox_xmin < 0 OR bbox_ymin < 0;
```

Listing 7: Indexes used to accelerate analytical queries

```
CREATE INDEX idx_annotation_image
    ON Annotation(image_id);

CREATE INDEX idx_annotation_category
    ON Annotation(category_id);

CREATE INDEX idx_image_dataset_split
    ON Image(dataset_id, split);
```

A.3 Additional Figures

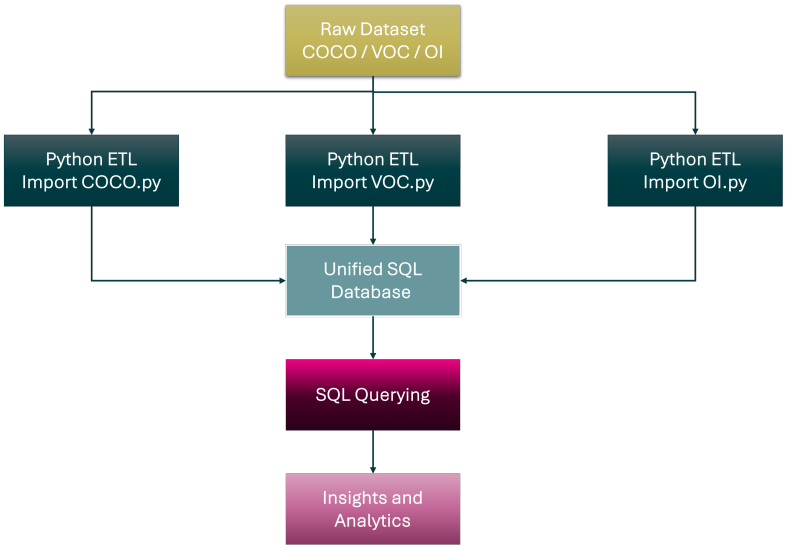


Figure 1: System pipeline for unifying COCO, VOC, and OpenImages using dataset-specific Python ETL scripts. Each dataset is independently parsed and normalized before being inserted into the unified SQL database, which supports downstream querying and analytics.

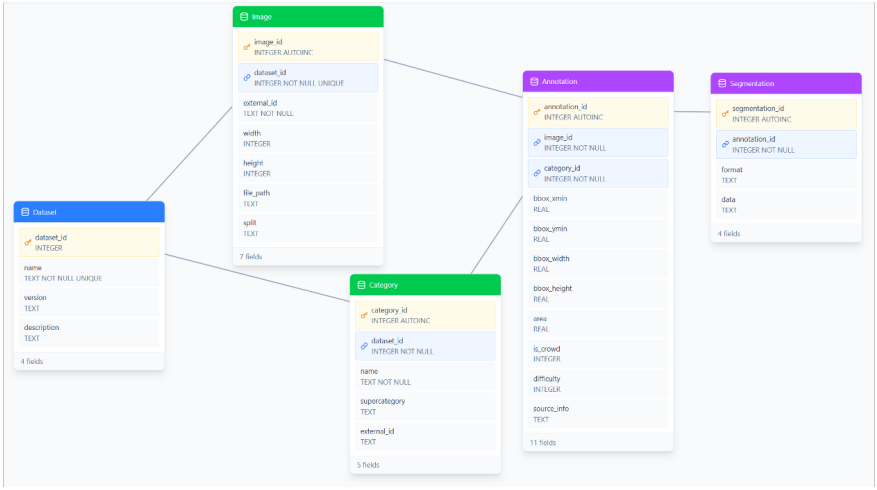
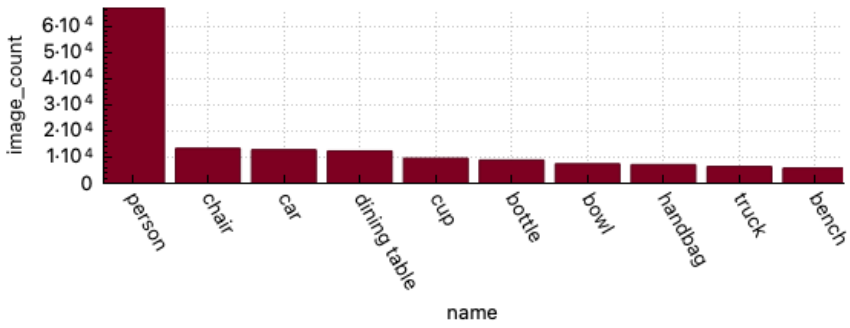
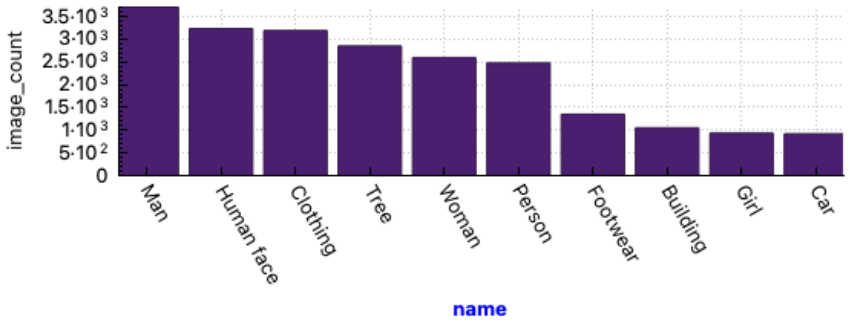


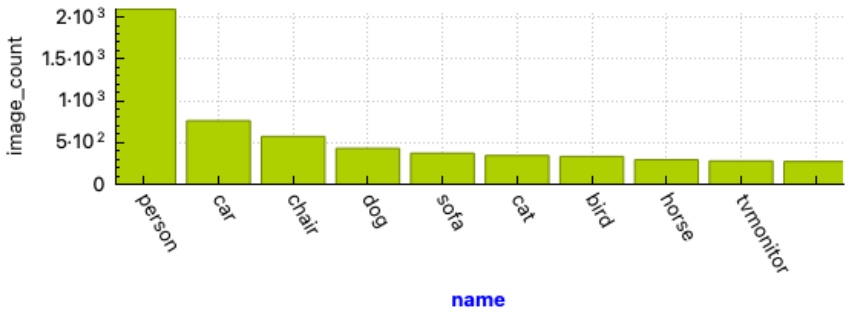
Figure 2: Entity–relationship diagram for the unified database schema. Five core tables represent datasets, images, categories, bounding-box annotations, and segmentations, with referential integrity enforced via foreign keys.



(a) COCO



(b) OpenImagesV7



(c) VOC 2007

Figure 3: Top 10 categories by image frequency per dataset.

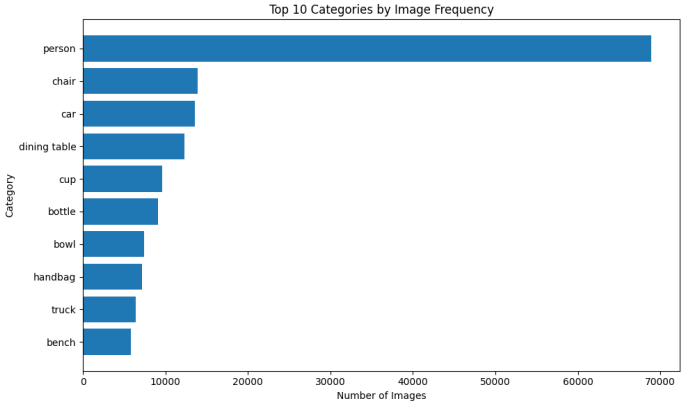


Figure 4: Top-10 object categories by number of images (COCO). The dataset exhibits strong long-tailed imbalance, dominated by the *person* class.

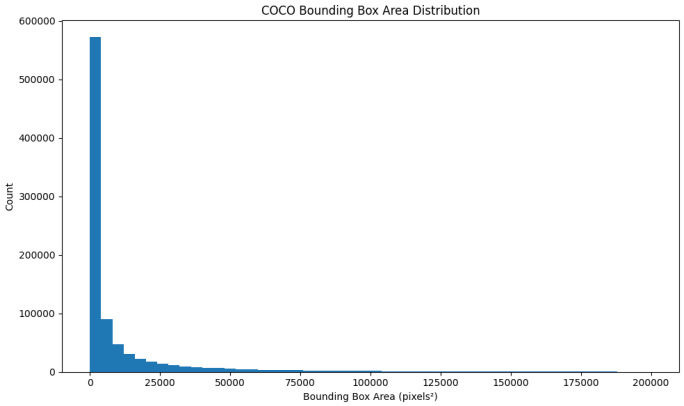


Figure 5: Distribution of bounding-box areas in COCO. The dataset contains a large concentration of small objects and a long tail of large objects.

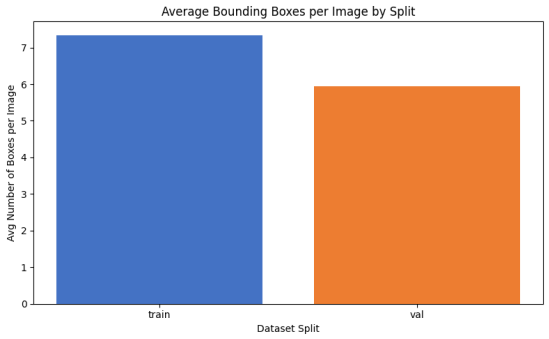


Figure 6: Average number of annotations per image across dataset splits. The COCO training split has substantially denser annotations than the validation split.

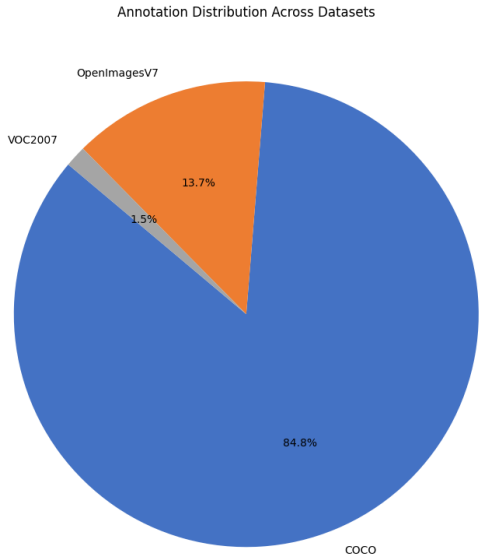


Figure 7: Proportional distribution of bounding-box annotations across datasets. COCO accounts for the majority of annotations in the unified system.

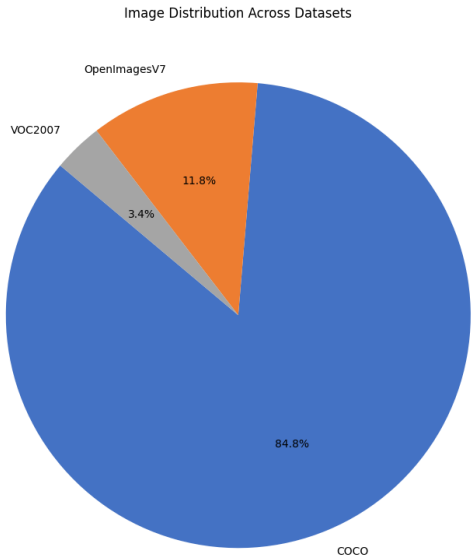
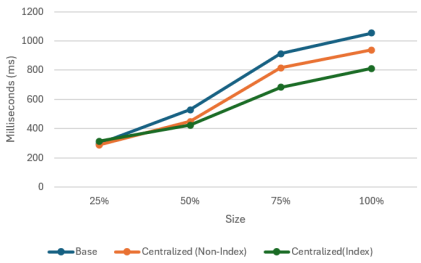
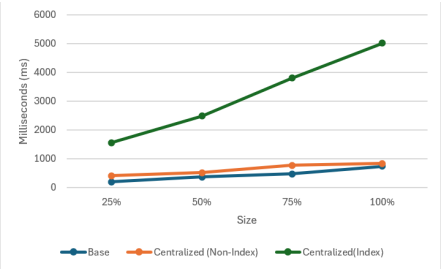


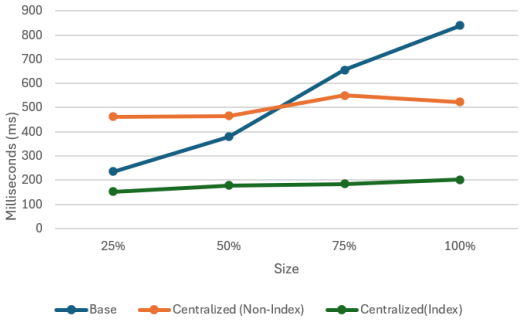
Figure 8: Proportional distribution of images across COCO, VOC, and OpenImages in the unified database.



(a) Listing 1



(b) Listing 3



(c) Listing 4

Figure 9: Average execution time by increasing dataset size per query.