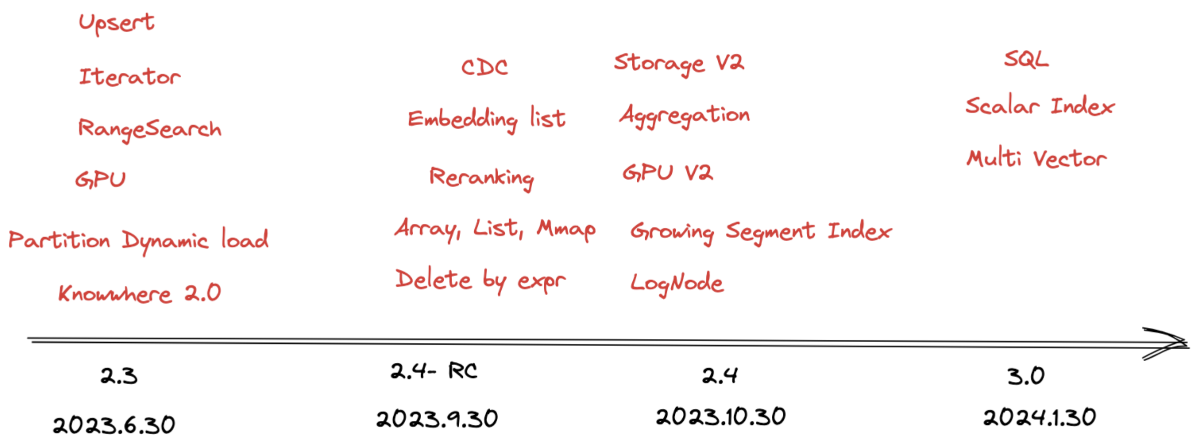
Feature plans



Features

	estimated deliver release	Urgency	Importance	Workload (month*person)	Details
SQL Support	2.4 Beta, 3.0 release	4	5	12	Support mysql connector, with insert, delete, search, aggregate, ddl support
Velox execution engine	2.3/2.4	4	4	6	Use velox to execute TableScan, Predicate, aggregation operators
MMap data management	2.4	3	4	3+	Load data into disk and mmap for searching. Let Milvus to serve data large than memory
Hybrid search with BM25 and vector	3.0 or later	2	4	6+	Search jointedly with bm25 score and vector distance score
Dynamic schema change	3.0	4	5	6+	Add, remove column
Distributed Log store	3.0 or later	2	3	6+	Implement distributed log device to replace kafka/pulsar for faster speed and recovery
Add Log Node and remove datanode	3.0	2	3	3+	Add log node to handle write/flush, datanode will merge with indexnode and only handle stateless jobs
Dynamic shard change	3.0 or later	2	2	3+	Change collection shard number in flight
Change data capture	2.3/2.4	3	3	3+	export inserted data to kafka and datawarehouse
Cluster level replication	2.4	4	4	3+	replicate data between two clusters for cross datacenter failure recovery
PITR	3.0 or later	1	2	3+	replay backup at any time
New persistent format	2.3	4	5	3+	Change bin log data format to improve search and recovery speed.
Ranking Support	3.0 or later	1	2	3+	Support complex ranking between scalar and vector score with machine learning model
Primary key dedup	3.0	4	4	3+	Dedup or overwrite when user write same primary key
Aggregation	2.3	5	4	3	Support count/groupby with where condition
Complex data type	3.0	2	4	3	Support list, set, json datatype and there queries such as IN
GPU	2.4/3.0	3	5	3	Support GPU based faiss and graph index
Multi vector support	3.0 or later	1	1	3	Need more user scenario
Condition delete	3.0	1	4	3	Delete from xxx where nonPK = ??
Fp16/Bf16 support	3.0	2	4	1+	Support BF16 and Fp16 could improve search latency and throught to 2X

Snapshot/Rollback	3.0 or later	1	1	3+	Snapshot is cool, but it's not as urgent for now
Support Quantization for graph index	2.4	4	4	1+	HNSW + PQ/SQ, NGT-PG
Auto Index 2.0	3.0	1	3	3+	Smart index parameter tuning
Support Models in Milvus	3.0 or later	1	4	6+	Support onnx models to do ranking and other models such as PCA
Data iterator	2.4	5	4	3	Iterate through all data with condition in the collection
Spark Connector	3.0	3	3	3	Combine spark to work with milvus together on offline processing
ScaNN Support	2.3	4	4	1+	Support scaNN in knowhere
Hedged Read	2.4	4	3	1+	when collection enable multiple replicas, hedged read helps to improve availability and reduce tail latency
Binary vector support	3.0	2	4	1+	Support binary vector in graph index
Support null data	2.4	2	4	3+	Support data to be null
Knowhere/Segcore metrics	2.4	5	5	3	Support prometheus based metrics collection
Vector as output field	2.4	3	4	1+	Support to retrieve vector field when search
Bulkload with clustering data	2.4	2	4	3	Support clustering data into segment before bulkload
Multi Vector	3.0	3	3	3	Support multiple vector field in single entity

Tools

Tracing	2.3	3	3	3	Dynamic tracing search/query request
WebUI	2.4	4	4	1+	Webui to show segment/channel distribution, index and collection stats
Milvus CLI	2.4	4	3	1	Help on triggering load balancing compaction, flush and other operations
Milvus system check	2.4	4	4	0.5	Check the consistency between etcd, S3 and memory
Backup	2.3	2	3	1	Back and restore data
performance diag tool	3.0 or later	1	1	1	diagnose performance , including cpu usage, memory usage and more
Health check	2.4	3	3	1	Check cluster health status

Other Enhancement

Hybrid search performance	2.3	3	5	3+	Improve search with filtering performance, especially for strict filtering condition such as PK=1
Streaming data search performance	2.3	5	5	3+	Improve search performance with concurrent write with read
Loadbalancing on large cluster	2.3	3	3	1+	Change current load balancing strategy
Failure recovery speed	2.3/2.4	4	4	3	Milvus can be fully recovered in 1 minuted under single machine crash, and zero down time with multiple replicas
Compaction optimization	2.4	4	4	3	1. Introduce major compaction to repartition data 2. refine minor compaction to handle frequent delete
Error code	2.4	5	4	3	Refine all error code and ensure each error returned has a correct error
access log	3.0 or later	1	1	1	record all the access log
Scalability	3.0	2	4	3	each shard can hold 1B data, test on 5B data set
LLM + Milvus DEMO	2.4	5	3	1+	A demo to show how Milvus can work together with openAI and huggingface
Memory control for flush, compaction and index building	2.4	3	4	3	ensure the memory utilization is stable when compaction and flush triggered.
Go, Java, Python, Cpp, NodeJs, Restful SDK refinement	2.3	5	4	1+	refine all sdk api and syncup,fully tested all the sdk listed
Build optimization	2.2/2.3	2	2	1+	Increase build speed, remove useless dependency, use conan as dependency management