Z5008: Big Data Engineering Master Reference

Big Data Lab (Labs 1–15) · Docker, Spark, Delta/Iceberg, Airflow, Kafka, MLlib, MLflow · innocent@iitmz.ac.in

1. Lakehouse Storage: Parquet, Delta Lake & Iceberg

- **Parquet Columnar Storage:** Row groups + column chunks + min/max page statistics enable predicate pushdown and column pruning. Target file size: 128–512 MB.
- **Delta Lake (`_delta_log`):** JSON commit logs (‘00000.json‘) + periodic Parquet checkpoints give Snapshot Isolation, ACID `MERGE INTO` upserts, and `VERSION AS OF` time travel.
- **File Compaction:** `OPTIMIZE table ZORDER BY (col)` compacts small files and co-locates multi-dimensional keys; `VACUUM` removes unreferenced files past retention.
- **Apache Iceberg:** Tracks columns by immutable **Field IDs** (safe rename/drop/add without rewrite) and supports hidden **partition evolution** across engines (Spark, Trino, Flink).

2. Apache Spark Execution & Tuning

- **Hierarchy:** Driver builds logical → physical plan via Catalyst; Stages split at **wide dependencies** (shuffles); Tasks run 1 per partition per executor core.
- **Narrow vs. Wide:** `filter`, `map`, `withColumn` (no shuffle) vs. `groupBy`, `repartition`, non-broadcast `join` (shuffle).
- **Join Strategies: Broadcast Hash Join** when smaller table $\leq$ `spark.sql.autoBroadcastJoinThreshold` (default 10 MB); **Sort-Merge Join** for large-to-large joins.
- **Skew Mitigation:** Enable Spark 3 **AQE** (`spark.sql.adaptive.enabled=true`, skew join + dynamic partition coalescing) or apply key **salting** ($k \mapsto (k, \text{rand}(0, S-1))$).
- **Spark SQL Windows:** `OVER (PARTITION BY k ORDER BY t ROWS BETWEEN 6 PRECEDING AND CURRENT ROW)`.

3. Airflow Orchestration & Workflow Guarantees

- **DAG Constraints:** Must be acyclic; scheduler runs tasks in topological order respecting `upstream >> downstream`.
- **Idempotency:** Key writes by logical execution date (`{{ ds }}`) using overwrite or `MERGE` so automatic retries never duplicate rows.
- **XComs:** Small metadata/URIs only (stored in Airflow metadata DB); pass large tables via object storage / Delta paths.
- **SLA & Alerts:** Track task duration against SLA budget; trigger callbacks on failure or SLA miss.

4. Kafka & Spark Structured Streaming

- **Kafka Ordering:** Total order guaranteed **within a single partition** (keyed by hash of message key); consumer groups scale up to 1 consumer per partition.
- **Delivery Semantics:** At-most-once, at-least-once (default, idempotent sink), exactly-once (`enable.idempotence=true` + transactional commits).
- **Structured Streaming Watermarks:** `.withWatermark("event_ti "2 minutes")` tracks $W = \max(\text{event_time}) - \text{delay}$, dropping late state cleanly after window close.
- **Output Modes: Append** (finalized watermarked windows), **Update** (changed rows), **Complete** (full aggregate table). Checkpoints persist offsets + RocksDB state store.

5. Distributed ML (MLlib / Ray) & MLflow Registry

- **MLlib Pipelines:** `Transformer.transform()` (deterministic, e.g., `VectorAssembler`) vs. `Estimator.fit()` (learns parameters, e.g., `StringIndexer`, `GBTClassifier`).
- **Ray Tune ASHA:** Asynchronous Successive Halving terminates underperforming trials early to cut hyperparameter search compute by 4–10×.
- **MLflow Lifecycle:** Log params, metrics, and model artifacts per `run_id`; register versions; enforce metric gates before transitioning `None` → `Staging` → `Production` (auto-archiving prior champion).

6. Serving, Observability & Governance

Metric / Tool	Role	Key Signal
Prometheus Counter	Monotonic total	`records_processed_tota`
Prometheus Gauge	Instant level	`kafka_consumer_lag`
Prometheus Histogram	Quantile buckets	`p99_latency_ms < 50`
BentoML + Docker	Model REST API	Micro-batching, `/health`
OpenLineage	Audit governance	Dataset-to-job provenanc

Drift Triage: Monitor **data drift** (input feature KL/PSI shift) alongside **concept/model drift** (online accuracy or calibration drop) and configure automatic canary rollback on SLA breach.

7. Production Lab Reference Patterns

- **Delta UPSERT Pattern:** `DeltaTable.forPath(spark, path).alias("t").merge(updates.alias("s"), "t.id = s.id").whenMatchedUpdateAll().whenNotMatchedInsertAll().exec`
- **Windowed Watermark Pattern:** `events.withWatermark("ts", "2 minutes").groupBy(window("ts", "5 minutes"), "page").count()`.
- **Salted Skew Join:** Expand skewed outer keys with random suffix `0..S-1` and explode matching small-table rows across all `0..S-1` buckets.
- **Container Readiness:** Separate `/healthz` (liveness) from `/readyz` (model artifact loaded in memory) before Kubernetes ingress routes traffic.

8. Cluster Sizing & Fault-Tolerance Rules

- **Executor Sizing:** Target 4–5 cores per executor and 128 MB partition sizes (`spark.sql.files.maxPartitionBytes`); reserve 10% container overhead memory for JVM off-heap buffers.
- **Broadcast Safety Guard:** Keep `autoBroadcastJoinThreshold` well below executor heap limits (default 10 MB; raise cautiously up to 100 MB only when driver and executors have headroom).

- **Kafka Consumer Lag Alert:** Alert when consumer lag increases monotonically across 3 consecutive Prometheus scrape intervals, indicating partition count or downstream sink saturation.
- **Lineage Auditability:** Emit OpenLineage `START/COMPLETE` events carrying dataset URIs, schema hashes, and Git SHA for reproducible regulatory audits.