Up-to-date Practice Test with Latest Questions and Answers covering latest syllabus and topics of the exam. Makes you ready to face actual exam.

KILL EXAMS

CCDAK Practice Questions
CCDAK Practice Test
CCDAK Practice Exam
CCDAK Exam Questions
CCDAK Study Guide

PASS

killexams.com

Confluent

CCDAK

Confluent Certified Developer for Apache Kafka

ORDER FULL VERSION

https://killexams.com/pass4sure/exam-detail/CCDAK

Question: 1703

CLI exploration for telecom call records, need to list, describe, and sample internal topics. Which commands?

A. LIST TOPICS EXTENDED;
B. DESCRIBE EXTENDED on collections
C. PRINT with INTERVAL for sampling
D. SHOW QUERIES for impacts

Answer: A,B,C
Explanation: Extended lists show internals. DESCRIBE details schemas. PRINT samples with controls.

Question: 1704

In a .NET producer application handling failed sends with retries exhausted, which callback method handles non-retriable errors like InvalidConfigException?

A. ProducerException in send future
B. OnAcknowledgment
C. DeliveryHandler with DeliveryReport.Error
D. flush() timeout

Answer: C
Explanation: Confluent.Kafka .NET client uses DeliveryHandler callback on Producer.Produce(). For errors after retries (non-retriable or exhausted), DeliveryReport.Error is set (e.g., RdKafka.ErrorCode._INVALID_CONFIG), allowing custom handling like logging or alerting.

Question: 1705

In a stateful streaming application, a compacted changelog topic backs a KTable for join operations. A consumer instance crashes without committing offsets, triggering a rebalance where partitions are reassigned. Upon restart, the application restores state from the changelog. What role does log compaction play in offset management and state restoration during this failure anomaly?

A. Uncommitted offsets lost on crash mean restoration starts from the last committed point, potentially incorporating compacted updates seamlessly.
B. Compaction ensures efficient state restoration by providing only the latest value per key from restored offsets onward.
C. If compaction removes records before restoration completes, state may become inconsistent.
D. Compaction bounds the restoration time by limiting historical data, relying on min.compaction.lag.ms for freshness guarantees.

Answer: A,B
Explanation: Compacted changelogs optimize KTable restoration by delivering only latest key values from the offset onward, reducing data volume and time for rebuilding state post-failure. Lost uncommitted progress on crash forces restart from committed offsets, but compaction integrates naturally, supplying the compacted tail for quick catch-up to current state. The process leverages compaction's key-based retention for compact, efficient recovery without full history replay.

Question: 1706

Handling exactly-once semantics in a sink connector processing transactional input topics.

A. Set exactly.once.semantics=true in connector config if supported (e.g., S3, GCS)
B. Use errors.tolerance=none with retries
C. Enable transactional.producer in worker
D. Configure producer overrides for idempotence

Answer: A
Explanation: Recent connectors (e.g., cloud storage) support connector-level exactly-once via pre-commit offset checks and atomic writes, preventing duplicates/at-least-once even on failures/retries, beyond basic idempotence.

Question: 1707

To explore managing consumer offsets in a transactional producer-consumer setup, which properties enable safe offset commits without risking inconsistency during failures?

A. Manual async commits only
B. transactional.id for consumer
C. isolation.level=read_committed with producer transactions
D. enable.auto.commit=true

Answer: C
Explanation: read_committed isolation ensures consumers see only committed transactional data, and including offsets in producer transactions (via sendOffsetsToTransaction)

guarantees atomicity, preventing partial reads or duplicates on failure.

Question: 1708

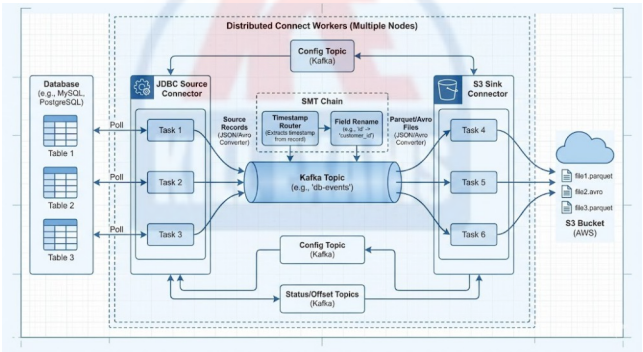
A Python consumer using confluent-kafka must handle graceful shutdown, committing final offsets and closing cleanly. What is the proper signal handling and API sequence?

A. Use consumer.poll(timeout=-1) blocking indefinitely.
B. Auto commit on close.
C. No commit needed on shutdown.
D. Register signal handlers for SIGTERM/SIGINT, in handler set a flag, break poll loop, consumer.commit(sync=True), then consumer.close().

Answer: D
Explanation: Signal handlers catch termination signals, set a shutdown flag checked in the poll loop to exit gracefully. Synchronous commit() ensures final offsets are durably stored before close(), which also leaves the group properly if applicable. This prevents offset loss or unnecessary rebalances, ensuring at-least-once processing resumption on restart in long-running Python consumers.

Question: 1709

Configuring Kafka Connect components for data flow from JDBC source to S3 sink with transformations.



Which component handles schema evolution compatibility?

A. Worker tasks directly
B. Connector plugins only
C. External SMTs
D. Converters and Schema Registry integration

Answer: D
Explanation: Converters (e.g., AvroConverter with Schema Registry) manage serialization/deserialization and enforce compatibility checks during data flow. Source/sink connectors use them to handle schema changes; SMTs can manipulate data but compatibility is enforced at conversion level with registry.

Question: 1710

You are developing a Kafka application that uses Avro serialization. Which of the following statements about the Schema Registry is correct?

A. It only stores schemas for Avro messages.
B. It provides a mechanism for managing schema versions.
C. It automatically generates schemas based on message content.
D. It requires all schemas to be registered before use.

Answer: B
Explanation: The Confluent Schema Registry provides a mechanism for managing schema versions, allowing developers to register, update, and retrieve schemas as needed. It supports multiple serialization formats, not just Avro, and does not automatically generate schemas.

Question: 1711

In complex scenario with producer metrics monitoring, which metric indicates batching efficiency?

A. batch-size-avg
B. buffer-total-bytes
C. record-retry-rate

D. metadata-age

Answer: A
Explanation: batch-size-avg shows average batch bytes; higher indicates better batching/ throughput.

Question: 1712

Topic 'sensor-data' configured segment.bytes=5MB, retention.bytes=50MB, delete.retention.ms=3600000. After burst of 100MB data, only 40MB remains despite time under 1hr. Why segments deleted?

A. Active segment compacted due to duplicates
B. delete.retention.ms preempts size policy
C. retention.bytes=50MB caps total partition size; oldest closed segments deleted first
D. log.roll.ms default forces early rollover

Answer: C
Explanation: Kafka enforces whichever retention limit hit first (time or size). retention.bytes applies cluster-wide per partition, deleting oldest closed segments beyond total. Time policy secondary. Compaction requires cleanup.policy=compact.

Question: 1713

You are using the ksqlDB CLI and want to see the current status of all your streams and tables. Which command should you execute?

A. `LIST STREAMS;`
B. `SHOW STREAMS;`
C. `SHOW TABLES;`
D. `DESCRIBE EXTENDED;`

Answer: B
Explanation: The command `SHOW STREAMS;` will display the current status of all streams in ksqlDB, including their names and other relevant metadata.

Question: 1714

You are developing a new feature for a messaging application that includes user preferences. The schema needs to evolve to accommodate additional preferences without breaking existing users. What should you do?

A. Remove old preference fields that are no longer relevant.
B. Provide default values for new fields in the schema.
C. Make all new preference fields optional.
D. Change existing preference fields to a different data type.

Answer: B,C
Explanation: Making new fields optional allows existing users to continue using the schema without issues. Providing default values for new fields ensures that consumers can handle messages even if they do not include the new preferences.

Question: 1715

In stock backtesting, Kafka Streams simulates replay with stream.filter((k,v)->v.timestamp > cutoff).mapValues(v->simulate(v)). Backtest diverges from prod. Cause?

A. Stream time vs event time
B. filter uses wall-clock
C. mapValues stateless
D. No repartition post-filter

Answer: B
Explanation: KStream.filter uses stream-time (max seen record time), dropping future event-time records in replays. Use Processor API or peek with event-time for accurate backtesting divergence fix.

Question: 1716

High difficulty: Combine idempotence and transactions.

A. Both for full EOS
B. transactional.id enables cross-session

C. Required together
D. Idempotence for single session

Answer: B,D
Explanation: Idempotence session-only; transactions with ID for persistence.

Question: 1717

What is the purpose of the `PARTITION BY` clause in a ksqlDB query?

A. To define the order of records in the output stream.
B. To specify the key for partitioning the output stream.
C. To filter records based on their partition assignments.
D. To group records based on a specific column for aggregation.

Answer: B
Explanation: The `PARTITION BY` clause is used to define the key for partitioning the output stream, which is essential for managing how data is distributed across Kafka partitions.

Question: 1718

You observe persistent under-replicated partitions in a healthy cluster despite sufficient disk space. Investigation reveals frequent preferred leader imbalances. What CLI command and configuration resolve this for balanced leadership?

A. Increase min.insync.replicas.
B. Set unclean.leader.election.enable=true.
C. kafka-reassign-partitions.sh for manual moves.
D. kafka-preferred-replica-election.sh --bootstrap-server localhost:9092 to manually trigger election, and set auto.leader.rebalance.enable=true for ongoing automatic balancing.

Answer: D
Explanation: Preferred replica election ensures leaders are on the first replica in assignment lists for optimal rack awareness and balance; manual CLI trigger corrects current imbalances, while enabling automatic leader rebalance schedules periodic checks and corrections, preventing sustained skew from maintenance or failures and distributing load evenly across brokers without risking data consistency.

Question: 1719

Stress testing idempotent producers at 1M TPS with enable.idempotence=true. Duplicates appear at 90% CPU.

A. Set retries=100000 with retry.backoff.ms=1
B. Tune batch.size=64KB and linger.ms=0
C. Monitor producer-metrics:request-latency-avg
D. Increase max.in.flight.requests.per.connection=1 to serialize

Answer: A,C
Explanation: Stress exposes idempotence limits; high retries/backoff handle retries without seq dupes; latency metrics pinpoint. In-flight >1 risks ordering; batch/longer for throughput.

Question: 1720

What is the purpose of Kafka Streams windowing operations?

A. To enable fault tolerance and high availability in Kafka deployments.
B. To manage and maintain the state of a Kafka Streams application.
C. To provide real-time monitoring and metrics for Kafka Streams applications.
D. To perform real-time windowed aggregations on data streams.

Answer: D
Explanation: Kafka Streams windowing operations allow you to perform real-time windowed aggregations on data streams. They enable you to group and aggregate data within specified time windows or based on other criteria, providing powerful stream processing capabilities for real-time analytics and computations.

Question: 1721

When configuring a Kafka producer to ensure that it can handle transient network issues gracefully, which properties should be set?

A. `linger.ms=100`
B. `max.in.flight.requests.per.connection=5`
C. `acks=0`
D. `retries=10`

Answer: A,B,D
Explanation: Setting `retries=10` allows the producer to attempt to resend messages if they fail due to transient issues. Configuring `max.in.flight.requests.per.connection=5` helps manage multiple requests in transit without losing order. Using `linger.ms=100` allows for some batching, which can also help mitigate network issues. Setting `acks=0` would not provide any guarantee of message delivery.

KILLEXAMS.COM

Killexams.com is a leading online platform specializing in high-quality certification exam preparation. Offering a robust suite of tools, including Exam Questions, practice tests, and advanced test engines, Killexams.com empowers candidates to excel in their certification exams. Discover the key features that make Killexams.com the go-to choice for exam success.

Practice Exam Questions Based on Current Exam Objectives

Killexams.com provides practice exam questions aligned with the latest official exam objectives and latest syllabus. Our content is reviewed and updated regularly to reflect recent changes announced by certification vendors. By studying these practice questions, candidates will cover the structure, difficulty level, and topics of the actual exam, helping them prepare more effectively and efficiently.

Comprehensive Practice Exams (PDF Format)

Killexams.com offers multiple-choice questions (MCQs) in easy-to-read PDF format, covering all major domains of the exam. Each PDF contains a structured collection of practice questions and verified answers designed to support focused study. These MCQs help candidates reinforce key concepts, identify knowledge gaps, and improve exam readiness through consistent practice.

Realistic Practice Tests (Online Test Engine & Desktop Test Engine)

To support hands-on preparation, Killexams.com provides practice tests through both an Online Test Engine and a Desktop Test Engine. These tools are designed to simulate a real exam environment, allowing candidates to practice under exam-like conditions, with latest syllbus and topics of the exam. Performance tracking, test history, and result analysis help users evaluate their progress and focus on areas that need improvement.

Risk-Free Purchase Policy

Killexams.com follows a transparent and customer-friendly purchase policy. If users are not satisfied with the study materials, they may request assistance or a refund in accordance with our published terms and conditions. This policy reflects our commitment to customer satisfaction, fairness, and confidence in our preparation resources.

Regularly Updated Content

Our practice question bank is reviewed and updated on an ongoing basis to stay aligned with the latest exam outlines and vendor updates. This ensures candidates are studying up-to-date, relevant material, and preparing with content that reflects current exam expectations, helping them stay confident and well-prepared.