Az adatfolyam pontos útja:

1. Producer → üzenet (routing key-jel) → exchange
2. Exchange → routing szabály alapján → sor (queue)
3. Sor → üzenet → consumer

A consumer és az üzenetek válogatása:

A consumer **nem** válogatja ki a neki szóló üzeneteket a routing key vagy header alapján. Ez a folyamat másképp működik:

- A consumer egy konkrét sorhoz kapcsolódik (feliratkozik rá).
- Az exchange-ek felelősek azért, hogy az üzeneteket a megfelelő sorokba irányítsák a routing key (direct exchange esetén) vagy más szabályok (topic, headers, fanout exchange-ek esetén) alapján.
- Amikor a consumer kapcsolódik egy sorhoz, akkor az abba a sorba irányított üzeneteket kapja meg, amelyekhez már előzőleg megtörtént a szűrés/irányítás az exchange által.

Tehát a szűrés alapesetben nem a consumer szintjén, hanem az exchange és a sorok közötti kötések (bindings) szintjén történik. A consumer csak azokat az üzeneteket látja, amelyek már bekerültek az általa figyelt sorba.

A fő megközelítési lehetőségek a feladat megoldására:

0. Megoldás: 1 Direct exchange, 1 Routing Key, 1 közös sor és 3 különböző header alapján szűrés

- Legegyszerűbb konfiguráció (1 exchange, 1 sor)
- A consumer az üzenet header mezői alapján dönti el, feldolgozza-e az üzenetet
- A szűrés teljes mértékben alkalmazás oldalon történik

1. Megoldás: 3 Direct Exchange, 3 Routing Key, 1 közös sor

- 3 különböző direct exchange, mind ugyanahhoz a sorhoz kötve
- Különböző routing key-ek használata
- A consumer a routing key alapján dönti el, feldolgozza-e az üzenetet

2. Megoldás: 1 Topic Exchange, 3 különböző routing pattern, 1 közös sor

- 1 topic exchange, különböző routing key mintákkal
- A legrugalmasabb megoldás
- A consumer a routing key mintája alapján dönti el, feldolgozza-e az üzenetet

Mindhárom megoldás érvényes és működőképes a feladatodra. A választás a rendszered többi részétől és a konkrét igényektől függ:

- A 0. megoldás a legegyszerűbb konfigurációt igényli
- A 2. megoldás a legrugalmasabb, ha később bővíteni szeretnéd a rendszert
- Az 1. megoldás akkor hasznos, ha az üzeneteket logikailag is külön akarod választani

1. Megoldás: 3 Direct Exchange, 3 Routing Key, 1 közös sor

Ebben a felállásban:

- Van 3 különböző direct exchange
- Mindegyik exchange a saját egyedi routing key-ével köti a közös sort
- A consumerek a kapott üzeneteket a routing key alapján tudják szűrni

```
# Producer oldalon
# Különböző producerek különböző exchange-eket használnak
channel.basic_publish(
    exchange='exchange_1',
    routing_key='key_1',
    body='Üzenet az 1-es consumernek'
)

# Másik producer
channel.basic_publish(
    exchange='exchange_2',
    routing_key='key_2',
    body='Üzenet a 2-es consumernek'
)

# Consumer oldalon
def callback(ch, method, properties, body):
    # A method.routing_key tartalmazza a használt routing key-t
    if method.routing_key == 'key_1':
        # Csak az 1-es consumer dolgozza fel
        print(f"1-es consumer feldolgozza: {body}")
        ch.basic_ack(delivery_tag=method.delivery_tag)
    else:
        # Nem ennek szól, visszautasítjuk
        ch.basic_reject(delivery_tag=method.delivery_tag, requeue=True)

# Kapcsolódás a közös sorhoz
channel.queue_declare(queue='shared_queue')

# Kösse össze mindhárom exchange-et ugyanahhoz a sorhoz
channel.queue_bind(exchange='exchange_1', queue='shared_queue',
routing_key='key_1')
channel.queue_bind(exchange='exchange_2', queue='shared_queue',
routing_key='key_2')
channel.queue_bind(exchange='exchange_3', queue='shared_queue',
routing_key='key_3')

# Fogyasztás
channel.basic_consume(queue='shared_queue', on_message_callback=callback)
```

2. Megoldás: 1 Topic Exchange, 3 különböző routing pattern, 1 közös sor

Ez az elegánsabb megoldás:

- Egyetlen topic exchange

- 3 különböző routing key pattern
- Mind ugyanahhoz a közös sorhoz kötve
- A consumerek a routing key alapján szűrnek

```
# Létrehozunk egy topic exchange-et
channel.exchange_declare(exchange='messages_topic', exchange_type='topic')

# Létrehozzuk a közös sort
channel.queue_declare(queue='shared_queue')

# Kötjük a sort az exchange-hez különböző routing key mintákkal
channel.queue_bind(exchange='messages_topic', queue='shared_queue',
routing_key='consumer.1.*')
channel.queue_bind(exchange='messages_topic', queue='shared_queue',
routing_key='consumer.2.*')
channel.queue_bind(exchange='messages_topic', queue='shared_queue',
routing_key='consumer.3.*')

# Producer oldalon
channel.basic_publish(
    exchange='messages_topic',
    routing_key='consumer.1.info',  # Ez az 1-es consumernek szól
    body='Üzenet az 1-es consumernek'
)

# Másik producer
channel.basic_publish(
    exchange='messages_topic',
    routing_key='consumer.2.warning',  # Ez a 2-es consumernek szól
    body='Üzenet a 2-es consumernek'
)

# Consumer oldalon
def callback_consumer_1(ch, method, properties, body):
    # Ellenőrizzük a routing key-t
    if method.routing_key.startswith('consumer.1.'):
        print(f"1-es consumer feldolgozza: {body}")
        ch.basic_ack(delivery_tag=method.delivery_tag)
    else:
        # Nem ennek szól, visszautasítjuk
        ch.basic_reject(delivery_tag=method.delivery_tag, requeue=True)

# Consumer feliratkozás a közös sorra
channel.basic_consume(queue='shared_queue',
on_message_callback=callback_consumer_1)
```

A Topic Exchange különösen jó megoldás

A topic exchange használata ad legnagyobb rugalmasságot, mert:

1. Csak egy exchange-et kell kezelned

2. A routing key minták segítségével nagyon rugalmasan szabályozhatod, melyik üzenet melyik consumerhez kerüljön
3. A routing key részből azonnal láthatod, kinek szól az üzenet

Mindkét esetben a consumer a `method.routing_key` értékét tudja ellenőrizni, és csak a neki szóló üzeneteket dolgozza fel. A többi üzenetet visszautasíthatja, hogy más consumerek feldolgozhassák.

Fontos: Ha több consumer is figyeli ugyanazt a sort, akkor gondoskodni kell róla, hogy minden üzenet csak a megfelelő consumer által legyen feldolgozva, különben az üzenetek elakadhatnak a rendszerben. Ez megoldható például azzal, hogy minden consumer explicit módon visszautasítja a nem neki szóló üzeneteket.