KIP-845: 'HasField' predicate for kafka connect

Status

Current state: *Discarded*

Discussion thread: *here*

JIRA: here

Please keep the discussion on the mailing list rather than commenting on the wiki (wiki discussions get unwieldy fast).

Motivation

Today's connect predicates enables checks on the record metadata. However, this can be limiting considering **many inbuilt and custom transformations that we have are more key/value-centric**.

Some use-cases this can solve:

- Data type conversions of certain pre-identified fields for records coming across datasets only if those fields exist. [Ex: `TimestampConverter` can be run only if the specified date field exists irrespective of the record metadata]
- Skip running certain transform if a given field does/does not exist. A lot of inbuilt transforms raise exceptions (Ex: `InsertField` transform if the field already exists) thereby breaking the task. Giving this control enable users to consciously configure for such cases.
- Even though some inbuilt transforms explicitly handle these cases, it would still be an unnecessary pass-through iteration.
- Considering each connector usually deals with multiple datasets (Even 100s for a database CDC connector), metadata-centric predicate checking will be somewhat limiting when we talk about such pre-identified common fields in the records from across the datasets.

Public Interfaces

New:

- *org.apache.kafka.connect.transforms.predicates.HasField$Key*
- *org.apache.kafka.connect.transforms.predicates.HasField$Value*

Proposed Changes

Overview

This KIP proposes to have a new HasField predicate class that works for records with or without schema.

- The predicate takes a field path configuration option string (Ex: `abc.xyz`). Hence supports nested field checks.
- Return true/false based on whether the path exists in the record.

Configuration changes

Predicate Settings in Kafka Connect Config:

- *predicates.<predicateName>.field.path*

Examples

- Converting a timestamp only if `created_at` field exists

```
"transforms": "modifyDate",
"transforms.modifyDate.type": "org.apache.kafka.connect.transforms.TimestampConverter$Value",
"transforms.modifyDate.target.format": "yyyy-MM-dd",
"transforms.modifyDate.target.type": "string",
"transforms.modifyDate.field": "created_at",
"transforms.modifyDate.predicate": "hasCreatedAt",
"predicates": "hasCreatedAt",
"predicates.hasCreatedAt.type": "org.apache.kafka.connect.transforms.predicates.HasField$Value",
"predicates.hasCreatedAt.field.path": "created_at"
```

- Inserting only if the field `testField` doesn't exist

```
"transforms": "insertTestField",
"transforms.insertTestField.type": "org.apache.kafka.connect.transforms.InsertField$Value",
"transforms.insertTestField.static.field": "testField",
"transforms.insertTestField.static.value": "test val",
"transforms.insertTestField.predicate": "hasTestField",
"transforms.insertTestField.negate": "true",
"predicates": "hasTestField",
"predicates.hasTestField.type": "org.apache.kafka.connect.transforms.predicates.HasField$Value",
"predicates.hasTestField.field.path": "testField"
```

Compatibility, Deprecation, and Migration Plan

Fully respects the current predicate interfaces and there is no need for deprecation/migration.

Test Plan / Cases

1. `field.path` setting is mandatory when the predicate is enabled
2. The predicate will return false for records with null keys/values.
3. The predicate returns true if the field exists in the record, even if the field is set to null.

Rejected Alternatives

- Creating multiple custom transforms to handle the field checks.