apache superset 에서 impala database 를 선택하고 query 에서 show tables in DB like '%find_table%';
이렇게 하고 싶은데

Superset에서 **Impala** 커넥션을 잡으면 SQL Lab에서 실행하는 쿼리는 결국 Impala가 직접 받아서 실행해요.
문제는 말씀하신

```sql
sql

show tables in DB like '%find_table%';
```

이 구문이 **Impala SQL 문법**에서는 지원되지 않는다는 점이에요.

- Hive에서는 **`SHOW TABLES IN db LIKE 'pattern'`** 구문이 되지만,
- Impala에서는 **`SHOW TABLES`**까지만 되고 **`LIKE`** 절은 안 붙습니다.

Impala에서 가능한 방법

1. **DB 지정 후 테이블 패턴 필터링**

   ```sql
   sql

   show tables in DB;
   ```

   → 전체 테이블을 가져온 뒤, Superset의 결과 화면에서 검색창으로 필터링.

2. **`information_schema` 조회 활용**
   Impala 3.0+부터는 **`information_schema.tables`** 뷰가 있어서 SQL로 필터링이 가능해요.

   ```sql
   sql

   SELECT table_name
   FROM information_schema.tables
   WHERE table_schema = 'DB'
     AND table_name LIKE '%find_table%';
   ```

3. **Hive Metastore 쿼리 (만약 Hive와 연동된다면)**
   Impala 자체는 제한이 있어서, 필요하다면 Hive CLI 쿼리나 metastore DB에서 직접 검색하는 방식도 씀.

👉 정리하면 Superset에서는 **SQL Lab 쿼리 창에서 2번 방식 (`information_schema.tables`)** 쓰는 게 가장 깔끔합니다.

혹시 지금 Impala 버전 알 수 있을까요? (3.0 이상이면 **`information_schema`** 접근이 가능하고, 그 전이면 **`show tables`** 결과를 수동으로 필터링해야 합니다.)

> impala query 로 버전 체크하는것?

Impala에서 버전 확인하려면 두 가지 길이 있어요.

1. SQL 쿼리로 확인

Impala 자체는 **`select version();`** 같은 함수가 없어요. 대신 **`SET` 명령**으로 서버 프로퍼티를 보면 됩니다:

```sql
sql

SET;
```

→ 출력 중에 이런 키들이 보일 거예요:

```pgsql
pgsql

impala.version
impala.build.version
impala.build.hash
```

2. Shell 명령 (beeline, impala-shell)

터미널에서 **`impala-shell`** 들어가면 접속 시점에 배너 메시지로 버전이 찍힙니다:

```cpp
cpp

Starting Impala Shell using Impala 4.1.0 ...
```

3. HTTP API

운영 환경이라면 Impala Daemon (**`impalad`**) 웹 UI → **`http://<impala-host>:25000/`**
→ **"Build Version"** 항목에서 확인 가능.

📌 Superset SQL Lab에서만 확인하려면 → **`SET;`** 실행해서 **`impala.version`** 값을 보는 게 제일 확실합니다.

원하시는 건 Superset에서 바로 확인할 수 있는 방법일까요, 아니면 클러스터 자체에서 버전 파악하는 게 필요하신 걸까요?

> show tables 해서 filter 할 수 없어?

Impala는 Hive처럼

```sql
sql

SHOW TABLES IN db LIKE 'pattern';
```

이런 걸 바로 지원하지 않습니다. **`SHOW TABLES`** 결과는 그냥 전부 찍힐 뿐이라 SQL 레벨에서 **`WHERE`** 같은 필터링이 안 돼요.

그래서 방법은 두 가지예요:

---

1. `information_schema` 활용 (Impala 3.0+)

버전이 3.0 이상이면 이렇게 가능합니다:

```sql
sql

SELECT table_name
FROM information_schema.tables
WHERE table_schema = 'db'
  AND table_name LIKE '%find_table%';
```

→ 이게 사실상 Hive의 **`SHOW TABLES ... LIKE`** 대체.

---

2. `SHOW TABLES` + 클라이언트 필터

Impala 2.x처럼 구식이면:

```sql
sql

SHOW TABLES IN db;
```

→ 결과 다 가져온 뒤에 Superset의 **SQL Lab 결과창 검색박스**에서 **`find_table`** 입력해서 필터링.
SQL 차원에서가 아니라 클라이언트에서 후처리하는 방식이죠.

👉 정리하면:

- **Impala 3.0 이상** → **`information_schema.tables`** 쿼리로 필터 가능.
- **그 이하** → **`SHOW TABLES`**만 가능, LIKE 불가.

원하시는 건 Superset에서 쿼리로 바로 걸러서 보시는 거죠?

> information_schema.tables 쿼리가 안되는데.

가능해요. 다만 **Impala는 `LIKE`에서 `%`가 아니라 `*` 와일드카드**를 씁니다.
(그리고 **`LIKE`** 키워드는 선택사항이에요.)

예시 둘 중 편한 걸로 쓰세요:

```sql
sql

-- 1) DB를 지정해서 한 방에
SHOW TABLES IN db_name LIKE '*find_table*';
```

```
-- 2) 먼저 DB 진입 후
USE db_name;
SHOW TABLES '*find_table*';
```

팁

- 여러 패턴 OR:

  ```
  sql

  SHOW TABLES IN db_name LIKE '*sales*|*order*';
  ```

- Impala는 오브젝트 이름을 **소문자**로 저장하니, 패턴도 소문자로 쓰는 게 안전합니다.
- `%` 대신 반드시 `*` 사용하세요. `%`는 매칭 안 됩니다.

공식 문법에 그대로 있어요: **`SHOW TABLES [IN db] [[LIKE] 'pattern']`** 이고 패턴은 **유닉스 스타일** `*` `+` `|` 입니다.
impala.apache.org

`information_schema.tables`가 안 되는 건 정상일 수 있어요(Impala 환경/버전에 따라 미지원). 위 **`SHOW TABLES ... LIKE`**가 Superset SQL Lab에서 가장 간단한 해결책입니다.

> 너 왜 방금전에는 안된다고 한거니?

맞습니다. 제가 처음엔 Hive 습관대로만 보고 "Impala는 **`SHOW TABLES LIKE`** 안 된다"고 단정해버린 거예요. 사실 Impala는 되긴 되는데, `%` **대신** `*` 와일드카드를 쓰는 식으로 문법이 달라요.

즉 Hive:

```
sql

SHOW TABLES IN db LIKE '%find_table%';
```

Impala:

```
sql

SHOW TABLES IN db LIKE '*find_table*';
```

제가 초반에 Hive랑 헷갈려서 잘못 드린 거고, 뒤에 정리하면서 바로잡은 거예요.
— 그래서 지금은 Impala 쿼리에서 `*` 와일드카드 쓰시면 원하는 대로 필터링 됩니다.

👉 혹시 Superset SQL Lab에서 바로 이 쿼리 돌려보셨을 때 에러가 뜨던가요, 아니면 결과가 빈 리스트로 나온 건가요?

> 자, 본론으로 들어가서 superset에서 impala로 해서 table 목록 확인하고 query 하면 정상적으로 되는데, starrocks 에서는 set catalaog hive; 하고 나서 테이블 목록을 조회하면 특정 테이블 몇개가 안나오거든, 이런건 왜 그런거야? 그리고 어떻게 디버깅 할 수 있지?

좋아요, 증상 정리부터 할게요.
Impala에서는 보이는데 StarRocks(Hive catalog)에서는 몇 개 테이블이 목록에 안 뜬다—이건 보통 아래 네 가지 축에서 걸려요.

왜 누락되나 (가능성 높은 원인 Top 6)

1. **지원하지 않는 테이블/스토리지 핸들러**
   Hive 메타스토어에 등록돼 있어도, StarRocks가 지원하지 않는 유형이면 목록/조회에서 빠질 수 있어요.
   - 예: Kudu 테이블, 커스텀 SerDe, 특수 스토리지 핸들러 등. FE 로그에 **`KuduMetadata.getTable`** 같은 흔적이 있었다면 특히 의심됩니다.
   - StarRocks의 데이터 레이크 지원 범위(파일포맷·타입 제약)는 공식 문서에 정리돼 있어요. 일부 타입/포맷은 미지원입니다. docs.starrocks.io
2. **Hive View / 정의 파싱 이슈**
   StarRocks 3.1+부터 Hive View 조회를 지원하지만, View 정의를 StarRocks/Trino 구문으로 파싱 못 하면 실패합니다(보이거나, 보이더라도 조회 시 에러). docs.starrocks.io
3. **메타데이터 캐시 갱신 지연**
   FE가 Hive 메타데이터를 캐시합니다. 주기적 갱신을 켠 경우도 일정 주기로만 동기화돼요. 즉시 반영이 필요하면 수동 갱신이 필요합니다. docs.starrocks.io+1
4. **권한(Privilege) 문제**
   외부 카탈로그에 **USAGE** 권한이 없으면 **`SHOW TABLES`**에서 안 보입니다(쿼리는 우회 경로로 때때로 되는 경우가 있어 헷갈릴 수 있어요). docs.starrocks.io+1
5. **파일 포맷/데이터 타입 제약**
   TEXT에서의 MAP/STRUCT, INTERVAL/BINARY/UNION 등 몇몇 타입은 미지원이라 테이블 취급이 제한될 수 있어요. docs.starrocks.io
6. **스토리지 접근/설정(특히 MinIO/S3) 미스매치**
   동일 HMS라도 버킷, endpoint, path-style, 인증키가 일부 테이블에서만 달라지면 StarRocks가 안전하게 노출을 못 할 수 있습니다(목록 자체는 메타데이터만 보지만, StarRocks는 내부적으로 안전성 체크를 할 때가 있어요). 관련 속성은 카탈로그 생성 문서들을 참고해서 정합성 확인이 필요합니다. docs.starrocks.io+1

바로 해볼 디버깅 체크리스트 (순서대로)

1. **누락 테이블의 "정체"부터 확인 (Hive/Impala 쪽)**
   - Hive/Impala에서:

     ```
     sql

     DESCRIBE FORMATTED db.table_name;  -- 테이블 타입, Storage Handler 확인
     ```

   - 결과가 **Kudu/특수 핸들러/커스텀 SerDe/Hive ACID 특성**을 쓰면 StarRocks 미지원 가능성 ↑
   - (StarRocks는 오픈 포맷 중심 지원: Hive/ORC/Parquet/Avro 등. 일부는 제약 있음) docs.starrocks.io
2. **StarRocks 권한 확인**

```sql
SHOW CATALOGS;
-- 해당 hive 카탈로그가 보이는지, 계정에 USAGE가 있는지 점검
```

필요 시 `GRANT USAGE ON CATALOG hive TO 'user'@'%';` 형태로 부여. docs.starrocks.io

3. **메타데이터 강제 갱신**
   - 특정 테이블만 즉시 갱신:

     ```sql
     REFRESH EXTERNAL TABLE hive.db.table_name;
     ```

     (권한: ALTER_PRIV 필요) docs.starrocks.io
   - 파티션 추가/삭제 직후 특히 필요합니다. docs.starrocks.io
   - 주기적 캐시 리프레시는 FE 설정으로 동작(미접속 카탈로그는 갱신 안 함). 운영값도 함께 확인. docs.starrocks.io+1
4. **목록에서 실제로 안 뜨는지 재현**

```sql
-- DB 지정해서 패턴으로 확인
SHOW TABLES FROM hive.db LIKE '*part_of_name*';
```

(StarRocks SHOW TABLES 문법과 외부 소스 접근 권한 주의) docs.starrocks.io

5. **View라면 간단한 SELECT로 파싱 여부 점검**

```sql
SELECT * FROM hive.db.view_name LIMIT 1;
```

파싱 실패 시 에러 메시지에 원인이 찍힙니다(함수/구문 호환성). docs.starrocks.io

6. **FE 로그 확인 (당신 환경 힌트: `KuduMetadata.getTable`)**
   - FE `fe.log`에서 해당 DB/테이블명으로 검색.
   - `Not supported`, `NoSuchObjectException`, `Permission denied`, `storage handler ...` 등의 메시지가 나오면 유형 판단이 빨라요.
7. **MinIO/S3 Catalog 속성 교차 확인**
   - (예시) endpoint, access/secret, path-style 사용 여부, 버킷/프리픽스 불일치가 없는지.
   - 카탈로그 정의 예시는 공식 카탈로그 문서(iceberg/hive/unified)에서 패턴 확인. docs.starrocks.io+1

현장용 빠른 분기표

- **Impala에서 Kudu 아니고, ORC/Parquet 일반 테이블인데만 누락**
  → ②권한, ③캐시, ⑦S3/MinIO 속성부터 보세요.

- **누락 대상이 전부 View**
  → ⑤에서 파싱 에러 확인(3.3.5에서 지원은 되지만 정의에 따라 실패). docs.starrocks.io
- **특정 타입만 빠짐(예: TEXT+MAP/STRUCT, 특수 타입 컬럼)**
  → ⑤/⑥와 함께 포맷·타입 제약 확인. docs.starrocks.io
- **FE 로그에 Kudu 관련 스택 흔적**
  → HMS에 Kudu 테이블이 섞여 있고 StarRocks 미지원 → "목록 미노출/조회불가"가 정상 동작일 수 있어요.

"전체 목록을 한 번에 갱신"하고 싶다면

StarRocks는 카탈로그/DB 단위 일괄 리프레시 명령이 제한적이라(문서 기준) 실무에선 **스크립트로 테이블별 REFRESH**를 돌립니다.

예시(클라이언트에서):

```
sql


-- 1) 목록 뽑기
SHOW TABLES FROM hive.db;

-- 2) 클라이언트(쉘/파이썬)에서 위 결과로 다음을 생성·실행
--    REFRESH EXTERNAL TABLE hive.db.t1;
--    REFRESH EXTERNAL TABLE hive.db.t2;
--    ...
```

(REFRESH는 Hive/Hudi용으로 명시돼 있습니다.) docs.starrocks.io

필요하면 지금 **문제가 되는 테이블 이름 1~2개만** 알려주세요.

- 제가 바로 **`DESCRIBE FORMATTED`에서 확인할 포인트**(tableType, storage handler, SerDe)랑
- **StarRocks에서 점검할 SQL/로그 키워드**를 딱 맞춰서 적어드릴게요.

환경 정보는 이미 기억해뒀습니다: RHEL K8s, MinIO, Helm/ArgoCD, StarRocks FE/CN 3.3.5, Operator 1.9.8.
이 조합이면 캐시/권한/스토리지 속성에서 많이 잡힙니다.

> 디버깅 할 수 있는 방법을 아주 상세히 정리해줘. 내가 step by step으로 단계적으로 디버깅 해볼께.

좋습니다. "Impala에서는 보이는데 StarRocks(Hive catalog)에서만 일부 테이블이 안 뜨는" 상황을 **현장에서 바로 돌려볼 수 있는 디버깅 플레이북**으로 정리했습니다.
(전제: K8s/RHEL, MinIO, StarRocks FE/CN 3.3.5, Operator 1.9.8)

---

0. 준비 (이슈 범위 좁히기)

- 누락된 **테이블 이름 2~3개**만 우선 픽스해 두세요.

- 각각에 대해 Impala/Hive에서 **`DESCRIBE FORMATTED`** 출력 캡처(테이블 타입, StorageHandler/SerDe, Location, Properties).

```sql

-- Impala/Hive(둘 다 가능)
DESCRIBE FORMATTED db_name.table_name;
```

---

1. StarRocks에서 재현 & 1차 확인

1. 카탈로그/DB/패턴 지정으로 "진짜 목록에서 안 보이는지" 확정

```sql

SET CATALOG hive;
SHOW DATABASES FROM hive LIKE '%db_name%';
SHOW TABLES FROM hive.db_name LIKE '%part_of_name%';
```

- 정말 목록에 없으면 다음 단계 진행.
- 목록에는 없지만 **직접 SELECT는 되는**(稀) 경우도 있어요. 아래도 확인:

2. **직접 조회 시도(바로 에러메시지 수집)**

```sql

SELECT * FROM hive.db_name.table_name LIMIT 1;
```

- 나오는 에러 문구(Unsupported, NoSuchObjectException, Permission denied 등)가 방향키가 됩니다. 메모해두세요.

---

2. FE 로그에서 결정적 단서 뽑기

1. FE 로그 스트리밍

```bash

# FE 파드, 컨테이너 이름은 환경에 맞게
kubectl get po -n <ns> -l app.kubernetes.io/name=starrocks-fe
kubectl logs -n <ns> <fe-pod> -c starrocks-fe -f | tee fe.log
```

2. 재현 시점에 로그 키워드로 그랩

```bash

# 대표 키워드
grep -E "Hive|metastore|NoSuchObject|Unsupported|Kudu|SerDe|Permission|S3|MinIO|AccessDenied"
fe.log
```

3. **로그-원인 매핑 예시**

- `... Kudu ... Unsupported ...` → Hive 메타스토어엔 존재하나 **Kudu 테이블**로 StarRocks 미지원.
- `NoSuchObjectException` → HMS 권한/이름 불일치/캐시 오래됨.
- `Permission denied / AccessDenied` → MinIO/S3 권한/정책/키 문제.
- `storage handler ... not supported` → 특수 StorageHandler/SerDe 미지원.
- `unsupported type ...` → MAP/STRUCT/UNION 등 타입 제약.

---

3. 메타데이터 캐시·목록 갱신

"전체 목록 리프레시"는 카탈로그 단위 일괄 명령이 제약 있어 **테이블별 REFRESH**를 쓰는 게 안전합니다.

1. 특정 테이블 즉시 갱신

```sql
REFRESH EXTERNAL TABLE hive.db_name.table_name;
```

2. **DB 전체를 스크립트로 일괄 REFRESH** (현장용)

```bash
# StarRocks FE(MySQL 프로토콜)로 목록 뽑아 한 번에 REFRESH
mysql -h <fe-host> -P 9030 -uroot -N -e "SET CATALOG hive; SHOW TABLES FROM hive.db_name;" \
| awk '{print "REFRESH EXTERNAL TABLE hive.db_name.`"$0"`;"}' \
| mysql -h <fe-host> -P 9030 -uroot
```

- 새 파티션/신규 테이블 반영 지연이 원인일 때 효과 확실합니다.

---

4. 권한(Privilege) 체크

외부 카탈로그에 USAGE/SELECT 권한이 없으면 `SHOW TABLES`에서 누락될 수 있습니다.

```sql
-- 내 권한 확인
SHOW GRANTS;
-- 필요 시 (예)
GRANT USAGE ON CATALOG hive TO 'user'@'%';
GRANT SELECT_PRIV ON EXTERNAL TABLE hive.db_name.table_name TO 'user'@'%';
```

- 사내가 Ranger/Sentry 정책을 쓰면, **Impala와 StarRocks에 적용되는 정책이 다를 수 있음**(중요 포인트).

---

5. Impala vs StarRocks "메타 차이" 비교

여기서 0단계 캡처가 힘을 발휘합니다.

1. **테이블 유형**

- Kudu / ACID(Transactional) / 뷰(View) / 외부(External) / 매니지드(Managed)
  - Kudu/ACID/특수 Handler는 StarRocks 미지원·제약 가능성이 큼.

2. **Storage Handler / SerDe / InputFormat**

- 예: `org.apache.hadoop.hive.kudu.KuduStorageHandler` → 미지원
- 커스텀 SerDe(사내용) → 파싱 실패 가능

3. **파일 포맷 & 스키마**

- Parquet/ORC/Avro는 대체로 OK지만 **특수 타입(MAP/STRUCT/UNION/BINARY/INTERVAL)** 사용 여부를 확인.

4. **Location/버킷**

- MinIO 경로가 **다른 버킷/프리픽스**로만 지정돼 있거나, 일부 테이블만 다른 엔드포인트/Region을 쓰면 SR 쪽 접근 검증에서 막힐 수 있음.

---

6. MinIO/S3 연결·정책 점검

> "Impala(하둡 에코) 쪽은 HDFS 커넥터, StarRocks는 S3 API 직결" 구조 차이 때문에 한쪽만 보이는 일이 잦습니다.

체크리스트

- 카탈로그 정의의 **endpoint / access_key / secret_key / path-style** 값 재확인
- MinIO **Bucket Policy**: 해당 경로 `s3:ListBucket`, `s3:GetObject` 허용 여부
- 네임스페이스에서 FE/CN가 지정한 **IAM/환경변수/시크릿**이 최신인지(ArgoCD 롤아웃 시 누락 주의)
- 테스트:

```
bash

# FE 파드에서 직접 S3 경로 접근 테스트(awscli 설치돼 있으면)
kubectl exec -n <ns> <fe-pod> -- sh -lc 'AWS_ACCESS_KEY_ID=... AWS_SECRET_ACCESS_KEY=...
AWS_EC2_METADATA_DISABLED=true aws --endpoint-url http://minio:9000 s3 ls s3://bucket/prefix/'
```

- 여기서 403/SignatureDoesNotMatch 나오면 SR도 동일하게 실패합니다.

---

7. View/특수 구문 호환성

- **VIEW**가 전부 빠지거나 일부만 빠지면, 뷰 정의(SQL)에서 SR 파서가 모르는 함수/힌트/서브쿼리 구조가 원인일 수 있습니다.
- 재현:

```
sql

SELECT * FROM hive.db_name.view_name LIMIT 1;
```

- 실패 시 FE 로그에 파싱 실패 이유가 비교적 명확히 남습니다.

---

8. “이 테이블만 빠진다” 케이스 별 처방

- **Kudu만 빠짐** → 정상(미지원). Impala에서만 사용하세요(또는 ETL로 Parquet 외부테이블 복제).
- **최근에 만든 파티션/테이블만 빠짐** → 3단계 REFRESH로 해결.
- **Text/CSV + 복합 타입 컬럼** → 스키마 단순화(타입 교체) 또는 Parquet/ORC 변환 권장.
- **특정 버킷/프리픽스만 빠짐** → 6단계 MinIO 정책/엔드포인트 비교로 해결.

---

9. 카탈로그 설정 빠르게 검증

SQL로 카탈로그 정의를 노출할 수 있으면 제일 쉽습니다.

```sql
sql

-- 가능할 경우:
SHOW CREATE CATALOG hive;
```

- 여기서 보이는 엔드포인트/옵션이 현재 의도와 다른지 확인
- 불가하면, FE 구성(ConfigMap/Secret)에서 카탈로그 생성 SQL 또는 환경 변수를 확인

---

10. 두 시스템 목록 차이 “정답셋” 만들기

현장용으로 확실히 하려면 **두 목록을 파일로 뽑아 차집합**을 구하세요.

```bash
bash

# Impala/Hive
impala-shell -q "show tables in db_name;" -B | tr -d '\r' | sort > /tmp/impala.txt

# StarRocks
mysql -h <fe-host> -P 9030 -uroot -N -e "SET CATALOG hive; SHOW TABLES FROM hive.db_name;" \
| sort > /tmp/sr.txt

comm -23 /tmp/impala.txt /tmp/sr.txt   # Impala엔 있는데 SR엔 없는 목록
comm -13 /tmp/impala.txt /tmp/sr.txt   # SR엔 있는데 Impala엔 없는 목록(레어)
```

---

11. 문제가 된 테이블만 집중 분석 (템플릿)

아래 템플릿으로 한 테이블씩 체크하면 결론 빨라집니다.

- ☐ Impala/Hive **`DESCRIBE FORMATTED db.t`** 결과에서
  Table Type/StorageHandler/SerDe/InputFormat/Location/transactional 확인
- ☐ StarRocks **`SELECT * FROM hive.db.t LIMIT 1;`** 에러문구 기록
- ☐ FE 로그에서 해당 테이블명/DB명으로 **`grep`**

- ☐ MinIO 버킷 경로 권한 테스트
- ☐ `REFRESH EXTERNAL TABLE hive.db.t;` 실행 후 재확인
- ☐ 권한(`SHOW GRANTS`)과 정책(Ranger/Sentry/MinIO Policy) 확인

---

12. 재발 방지 팁

- **신규/수정 테이블 자동 REFRESH**: 사내 파이프라인(메타 변경 시)에서 위 3-2 스크립트를 크론/잡으로 태우기
- **스키마 가드레일**: Kudu/ACID/커스텀 SerDe는 “Impala 전용” 태그를 붙여 문서화
- **MinIO 정책 표준화**: 카탈로그가 접근할 버킷/프리픽스에 공통 정책(읽기·목록) 적용
- **운영 점검 항목**: FE 로그에서 `metastore/Unsupported` 패턴 알람

필요하시면, **누락 테이블 이름 1~2개**만 알려주세요. 위 체크리스트 중 **어디부터 파고들지**를 그 테이블 기준으로 딱 맞춰 드릴게요.
지금 단계에선 2(로그) → 3(REFRESH) → 6(MinIO) 순으로 보시면 가장 빠르게 원인 나옵니다.