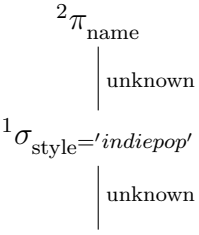
```
create table Person(
  id int primary key, [hash index of size 2000]
  dob date not null,
  name varchar(255) not null [tree index of size 1000]
);
create table Band(
  id int primary key, [hash index of size 1000]
  name varchar(255),
  style varchar(255)
);
create table Instrument(
  id int primary key, [hash index of size 200]
  name varchar(255) not null,
  musicalKey varchar(255)
);
create table plays(
  person int references Person(id),
  instrument int references Instrument(id),
  primary key(person, instrument) [tree index of size 4000]
);
create table memberOf(
  person int references Person(id),
  band int references Band(id),
  primary key(person, band) [tree index of size 4000]
);
create table StringInstrument(
  id int primary key references Instrument(id), [hash index of size 100]
  numberOfStrings int not null
);
create table BrassInstrument(
  id int primary key references Instrument(id), [hash index of size 50]
  numberOfValves int not null
);
create table PercussionInstrument(
  id int primary key references Instrument(id) [hash index of size 50]
);
```

- There are 2000 person records.
- There are 1000 band records.
- There are 200 instrument records.
- There are 4000 plays records.

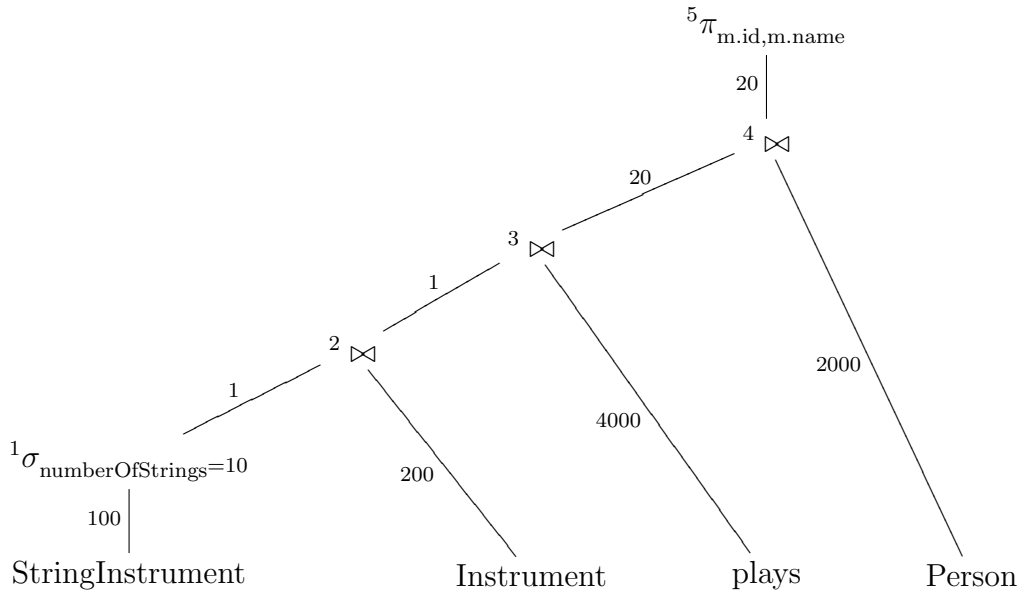
- So each person plays 2 instruments on average.
- Each instrument is played by 20 persons on average.
- There are 4000 memberOf records.
- So each person is a member of 2 bands on average.
- Each band has 4 members on average.
- There are 100 string instruments, or 0.5 of all instruments.
- The range of values for number of strings is larger than 100 so only 1 or less of instruments has one particular number of strings (assuming a uniform distribution).
- There are 50 brass instruments, or 0.25 of all instruments.
- There are 50 percussion instruments, or 0.25 of all instruments.
- So 0.125 of all instruments are both brass and percussion or 25 of them.

```
select b.name
  from Band b
 where b.style = 'indie pop';
```



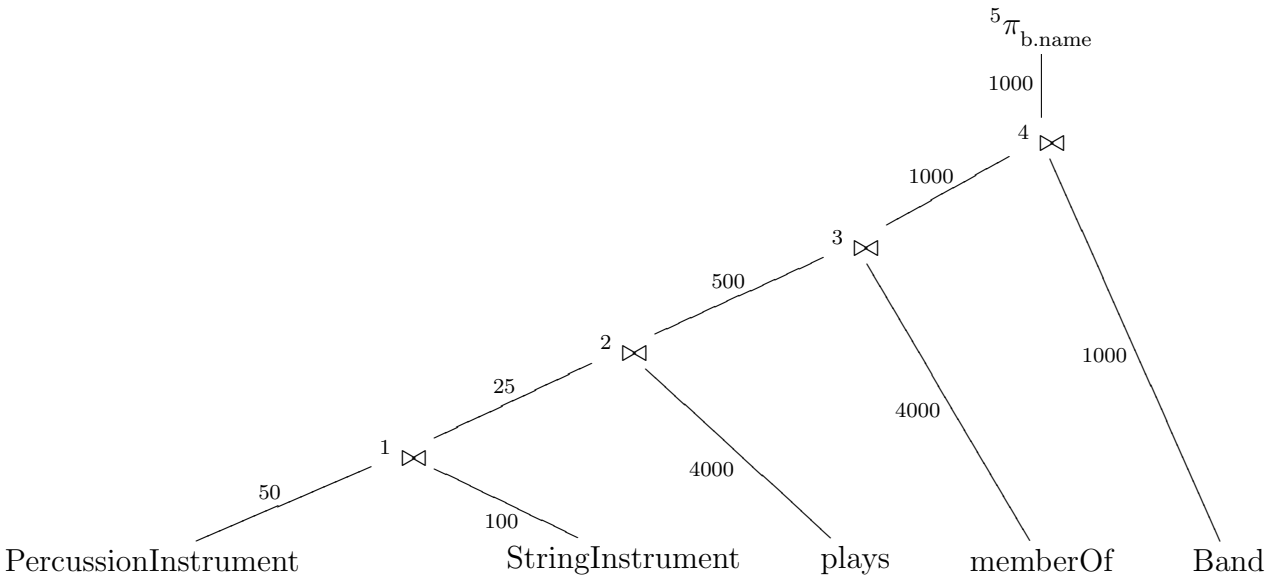
1. Scan. The style column is not indexed.
2. Scan. Duplicates are not eliminated, so no partitioning is necessary.

```
select m.id, m.name
  from Person m, plays p, Instrument i, StringInstrument si
 where p.person = m.id
   and p.instrument = i.id
   and i.id = si.id
   and si.numberOfStrings = 10;
```



1. Scan. The numberOfStrings column is not indexed.
2. Index nested loop. The target has an index and the number of retrievals is small.
3. Sort-merge. The target does not have a relevant index.
4. Index nested loop or sort-merge. This must be determined by computing the time for each strategy.
5. Scan. Duplicates are not eliminated, so no partitioning is necessary.

```
select b.name
  from Band b, plays p, StringInstrument si, PercussionInstrument pi, memberOf m
 where b.id = m.band
   and p.person = m.person
   and p.instrument = si.id
   and p.instrument = pi.id
```



1. Sort-merge. Target does not have a relevant index, but even if it did, all records are being joined so one should not use it.

2. Sort-merge. Target does not have a relevant index, but even if it did, the retrieval is not selective.

3. Sort-merge. Target does have a relevant index, but the retrieval is unselective so it should not be used.

4. Sort-merge. Target does have a relevant index, but the retrieval is unselective so it should not be used.

5. Scan. Duplicates are not eliminated, so no partitioning is necessary.