BIEN 3 - Task #884

fix Postgres bug that causes query planner to use seq scans and slow sorts instead of index scans in the import

04/02/2014 03:32 PM - Aaron Marcuse-Kubitza

Status:	Rejected	**Start date:**	04/02/2014
Priority:	High	**Due date:**	
Assignee:	Aaron Marcuse-Kubitza	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
Activity type:			

Description

issue

- see the following pg_stat_activity snapshots (note the EXPLAIN output below each query):

```
/*NVS.StemObservation*/    CREATE TEMP TABLE "stemobservation_pkeys" AS
SELECT
"in#11"."row_num"
, "stemobservation"."stemobservation_id" AS "out.stemobservation_id"
FROM "in#11"
JOIN "stemobservation" ON COALESCE("stemobservation"."height_m", CAST('NaN' AS double
precision)) = COALESCE("in#11"."NVS.StemObservation.Height::double", CAST('NaN' AS double
precision))
/* EXPLAIN:
Merge Join  (cost=1045217.21..95909185.15 rows=6323041431 width=8)
  Merge Cond: ((COALESCE(stemobservation.height_m, 'NaN'::double precision)) = (COALESCE("in#1
1"."NVS.StemObservation.Height::double", 'NaN'::double precision)))
  ->  Sort  (cost=30039.56..30492.76 rows=181279 width=12)
        Sort Key: (COALESCE(stemobservation.height_m, 'NaN'::double precision))
        ->  Seq Scan on stemobservation  (cost=0.00..14206.79 rows=181279 width=12)
  ->  Materialize  (cost=1015177.64..1050057.81 rows=6976033 width=12)
        ->  Sort  (cost=1015177.64..1032617.73 rows=6976033 width=12)
              Sort Key: (COALESCE("in#11"."NVS.StemObs

/*CVS.taxon_observation.***/    CREATE TEMP TABLE "location_pkeys" AS
SELECT
"in#5"."row_num"
, "location"."location_id" AS "out.location_id"
FROM "in#5"
JOIN "location" ON ("location"."accesslevel" = "in#5"."CVS.taxon_observation.**.accessRights::
accesslevel" OR ("location"."accesslevel" IS NULL AND "in#5"."CVS.taxon_observation.**.accessR
ights::accesslevel" IS NULL))
/* EXPLAIN:
Nested Loop  (cost=0.00..83049883712.45 rows=27683225000 width=8)
  Join Filter: ((location.accesslevel = "in#5"."CVS.taxon_observation.**.accessRights::accessl
evel") OR ((location.accesslevel IS NULL) AND ("in#5"."CVS.taxon_observation.**.accessRights::
accesslevel" IS NULL)))
  ->  Seq Scan on location  (cost=0.00..175778.45 rows=5536645 width=8)
  ->  Materialize  (cost=0.00..35434.00 rows=1000000 width=8)
        ->  Seq Scan on "in#5"  (cost=0.00..30434.00 rows=1000000 width=8)
*/

/*ARIZ.omoccurrences*/    CREATE TEMP TABLE "location_pkeys" AS
SELECT
"in#6"."occurrenceID"
, "location"."location_id" AS "out.location_id"
FROM "in#6"
JOIN "location" ON COALESCE("location"."elevation_m", CAST('NaN' AS double precision)) =
COALESCE("in#6"."_alt(1=verbatimElevation, 2=_avg(1=minimumElevat.result::double", CAST('NaN'
AS double precision))
/* EXPLAIN:
Merge Join  (cost=52039.20..3508151.98 rows=230343442 width=8)
  Merge Cond: ((COALESCE(location.elevation_m, 'NaN'::double precision)) = (COALESCE("in#6"."_
alt(1=verbatimElevation, 2=_avg(1=minimumElevat.result::double", 'NaN'::double precision)))
  ->  Sort  (cost=23780.86..24261.44 rows=192230 width=12)
        Sort Key: (COALESCE(location.elevation_m, 'NaN'::double precision))
        ->  Seq Scan on location  (cost=0.00..6910.30 rows=192230 width=12)
  ->  Sort  (cost=28258.34..28857.47 rows=239654 width=12)
        Sort Key: (COALESCE("in#6"."_alt(1=verbatimElevation, 2=_avg(1=minimumElevat.result::d
ouble", 'NaN'::double precision))
        -
*/
```

- this is separate from the disk space leak (#887), which is also associated with the above queries but caused by more than just the fact that it's using sorts

History

#1 - 04/02/2014 03:33 PM - Aaron Marcuse-Kubitza

- Description updated

#2 - 04/02/2014 03:37 PM - Aaron Marcuse-Kubitza

- Description updated

#3 - 04/02/2014 06:43 PM - Aaron Marcuse-Kubitza

- Description updated

#4 - 04/03/2014 09:02 AM - Aaron Marcuse-Kubitza

- Description updated

#5 - 04/11/2014 04:23 PM - Aaron Marcuse-Kubitza

- Description updated

#6 - 04/11/2014 04:23 PM - Aaron Marcuse-Kubitza

- Description updated

#7 - 04/17/2014 10:46 AM - Aaron Marcuse-Kubitza

- Description updated

#8 - 04/17/2014 10:52 AM - Aaron Marcuse-Kubitza

- Description updated

#9 - 04/23/2014 04:00 PM - Aaron Marcuse-Kubitza

- Description updated

- Status changed from New to Rejected

duplicate of #902: slow sorts are caused by joining on the wrong columns, not query planner settings