Item 149
git_comments:

git_commits:

1. **summary:** [SPARK-23907][SQL] Add regr_* functions
   message: [SPARK-23907][SQL] Add regr_* functions ## What changes were proposed in this pull request? The PR introduces regr_slope, regr_intercept, regr_r2, regr_sxx, regr_syy, regr_sxy, regr_avgx, regr_avgy, regr_count. The implementation of this functions mirrors Hive's one in HIVE-15978. ## How was this patch tested? added UT (values compared with Hive) Author: Marco Gaido <marcogaido91@gmail.com> Closes #21054 from mgaido91/SPARK-23907.

github_issues:

github_issues_comments:

github_pulls:

1. **title:** [SPARK-23907][SQL] Add regr_* functions
   body: ## What changes were proposed in this pull request? The PR introduces regr_slope, regr_intercept, regr_r2, regr_sxx, regr_syy, regr_sxy, regr_avgx, regr_avgy, regr_count. The implementation of this functions mirrors Hive's one in HIVE-15978. ## How was this patch tested? added UT (values compared with Hive)

github_pulls_comments:

1. **[Test build #89278 has finished](https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder/89278/testReport)** for PR 21054 at commit [`90c266e`](https://github.com/apache/spark/commit/90c266e09e276c37126d8428a4d00f4986d240e5). * This patch passes all tests. * This patch merges cleanly. * This patch adds the following public classes _(experimental)_: * `abstract class AverageAggregate extends DeclarativeAggregate ` * `case class Average(child: Expression) extends AverageAggregate with ImplicitCastInputTypes ` * `abstract class PearsonCorrelation(x: Expression, y: Expression)` * `case class Corr(x: Expression, y: Expression)` * `abstract class CountAggregate extends DeclarativeAggregate ` * `case class Count(children: Seq[Expression]) extends CountAggregate ` * `trait RegrBase extends AggregateFunction with ImplicitCastInputTypes ` * `case class RegrCount(y: Expression, x: Expression) extends CountAggregate` * `case class RegrSXX(y: Expression, x: Expression) extends CentralMomentAgg(x)` * `case class RegrSYY(y: Expression, x: Expression) extends CentralMomentAgg(y)` * `case class RegrAvgX(y: Expression, x: Expression) extends AverageAggregate` * `case class RegrAvgY(y: Expression, x: Expression) extends AverageAggregate` * `case class RegrSXY(y: Expression, x: Expression) extends Covariance(y, x)` * `case class RegrSlope(y: Expression, x: Expression) extends PearsonCorrelation(y, x)` * `case class RegrR2(y: Expression, x: Expression) extends PearsonCorrelation(y, x)` * `case class RegrIntercept(y: Expression, x: Expression) extends PearsonCorrelation(y, x)`
2. cc @ueshin
3. kindly ping @ueshin
4. I added the sql tests as you suggested @maropu. Doing this, I found that we have some difference in the way we return `NaN` and `null` wrt Hive. I haven't changed our behavior though, since this would introduce a behavior change for methods which already exist (corr). I think we have to discuss this and eventually fix in a separate PR if you agree, since this is not really related to the introduction of the new functions.
5. @mgaido91 I agree that the fix should be done in the separate PRs after the discussion. Could you list the differences you found for the future discussion? Thanks!
6. @ueshin I found that `corr` and `covar_samp` return `NULL` instead of `NaN` or `NaN` instead of `NULL`. So probably the way this cases are handled is different in Hive and Spark.
7. @mgaido91 Thanks for letting us know! cc @gatorsmile
8. @ueshin thank you for your attention.
9. **[Test build #90045 has finished](https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder/90045/testReport)** for PR 21054 at commit [`89295e6`](https://github.com/apache/spark/commit/89295e645e9aeea6ef7d47f35930215953a749d4). * This patch **fails Spark unit tests**. * This patch merges cleanly. * This patch adds no public classes.
10. **[Test build #90051 has finished](https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder/90051/testReport)** for PR 21054 at commit [`0d8cb46`](https://github.com/apache/spark/commit/0d8cb46ccf8cee7ba9d5c55d4cfef8f3011ece2c). * This patch **fails SparkR unit tests**. * This patch merges cleanly. * This patch adds no public classes.
11. @mgaido91 @ueshin Let us keep our behavior unchanged, although our outputs are different from Hive.
12. **[Test build #90121 has finished](https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder/90121/testReport)** for PR 21054 at commit [`e584adf`](https://github.com/apache/spark/commit/e584adfb972a4d56acd6f222f1c5a7affc1224a6). * This patch passes all tests. * This patch merges cleanly. * This patch adds no public classes.
13. LGTM, but I'd retrigger the build because it's been some time since the last build.
14. Jenkins, retest this please.
15. **[Test build #90442 has finished](https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder/90442/testReport)** for PR 21054 at commit [`e584adf`](https://github.com/apache/spark/commit/e584adfb972a4d56acd6f222f1c5a7affc1224a6). * This patch **fails due to an unknown error code, -9**. * This patch merges cleanly. * This patch adds no public classes.
16. Jenkins, retest this please.
17. **[Test build #90443 has finished](https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder/90443/testReport)** for PR 21054 at commit [`e584adf`](https://github.com/apache/spark/commit/e584adfb972a4d56acd6f222f1c5a7affc1224a6). * This patch passes all tests. * This patch merges cleanly. * This patch adds no public classes.
18. Thanks! merging to master.
19. There is not a single function that can’t be called by expr. It mainly adds some type safety. On Fri, May 11, 2018 at 7:18 PM Hyukjin Kwon <notifications@github.com> wrote: > *@HyukjinKwon* commented on this pull request. > ---------- > > In sql/core/src/main/scala/org/apache/spark/sql/functions.scala > <https://github.com/apache/spark/pull/21054#discussion_r187761743>: > > > @@ -775,6 +775,178 @@ object functions { > */ > def var_pop(columnName: String): Column = var_pop(Column(columnName)) > > + /** > + * Aggregate function: returns the number of non-null pairs. > + * > + * @group agg_funcs > + * @since 2.4.0 > + */ > + def regr_count(y: Column, x: Column): Column = withAggregateFunction { > > @rxin <https://github.com/rxin>, how about splitting up this file by the > group or something, or deprecating all the functions that can be called via > expr for 3.0.0? To me, it looked a bit odd when some functions exist and > some did not. It was an actual use case and I had to check which function > exists or not every time. > > — > You are receiving this because you were mentioned. > Reply to this email directly, view it on GitHub > <https://github.com/apache/spark/pull/21054#discussion_r187761743>, or mute > the thread > <https://github.com/notifications/unsubscribe-auth/AATvPMuMFZtp285MrttmJfITKM6WS0pcks5txkZ0gaJpZM4TSBOu> > . >

github_pulls_reviews:

1. Is it okay to remain as `override lazy val updateExpressions = updateExpressionsDef` as the same as other abstract classes?
2. How about declaring this as an argument of `AverageAggregate` as the same as other abstract classes?
3. nit: maybe we need line break before `extends`: ```scala case class RegrCount(...) extends CountAggregate with RegrBase { ... ``` , and ditto for the following functions.
4. nit: `Basee` -> `Base`?
5. We need to add tests in `sql/core/src/test/resources/sql-tests` for testing via the spark sql parser, too?
6. Since we always use `1e-8` for `checkAggregatesWithTol` in this test suite, how about setting this value as default in the function argument?
7. nit: we don't need tests for `null` in the left-side value?
8. How about `AverageLike` (`CountLike` `RegrLike`, too) along with `RankLike`? (It is just a suggestion)
9. ~How about `doUpdateExpressions`?~ I feel it'd be better to use a meaningful name for accumulating `sum` and` count`? e.g., `sumAndCountExpr`
10. `protected def`?
11. `protected def`?
12. nit: Remove the blank line (I know this is not related to this pr though...)
13. `abstract class`?
14. it needs to be a trait to be mixed in with the other abstract classes (`CountLike`, `AggregateLike`, `CentralMomentAgg`, ...)
15. nit: Would it be possible to use `org.apache.spark.sql.catalyst.expressions._`?
16. done
17. done
18. I don't think so, since we have this tested also in the SQL tests added.
19. nit: indent
20. nit: indent
21. Do we need `explain`?
22. we cannot because this would cause a compile error since there are two overloaded functions: ``` in class QueryTest, multiple overloaded alternatives of checkAggregatesWithTol define default arguments ```
23. Do we need to return `null` if `yMk === Literal(0.0)`?
24. ok, thanks!
25. Ah, ok.
26. I found it is usually to explain `REGR_SXX` by `REGR_COUNT(x, y) * VAR_POP(y)`, e.g., [1](https://www.ibm.com/support/knowledgecenter/en/SSPT3X_2.1.2/com.ibm.swg.im.infosphere.biginsights.bigsql.doc/doc/bsql_regr_sxx.html) and [2](https://docs.oracle.com/cd/B19306_01/server.102/b14200/functions132.htm). Is it better to follow it? Same for `REGR_SYY`, etc..
27. Here I am following Hive. This is Hive docs and it reflects how it is actually computed. I am not sure it is a good idea to change it, since we are not really computing it as `REGR_COUNT(x, y) * VAR_POP(y)`.
28. @gatorsmile What do you think?
29. It is reasonable to follow Hive. Personally, I like DB2 or Oracle, because normally these commercial dbms is more professional. :)
30. do we need all of these? seems like users can just invoke expr to do them. this file is getting very long.
31. @rxin, how about splitting up this file by the group or something, or deprecating all the functions that can be called via expr for 3.0.0? To me, it looked a bit odd when some functions exist and some did not. It was an actual use case and I had to check which function exists or not every time.

jira_issues:

1. **summary:** Support regr_* functions
   description: Support the standard regr_* functions, regr_slope, regr_intercept, regr_r2, regr_sxx, regr_syy, regr_sxy, regr_avgx, regr_avgy, regr_count. SQL reference section 10.9

jira_issues_comments:

1. [~pxiong] I see multiple ways this could be achieved...and I'm not sure which one to take :) Most of these functions (more/or less) could be translated into existing UDAF function usage - it needs some tweaking; but it can be done; I don't really want to reimplement all those things again - I think it would be better to reuse them. # if I create some 'cover' UDAF evaluators for each of these functions and do the evaluation of those inside the new evaluator - that could work; but it will be quite a few very similar classes # tho other alternative is to add some slightly extended versions of some existing UDAFs (like:count and variance) - and rewrite somehow the {{regr_sxx(y,x)}} invocations to {{extended_COUNT(x, y) * extended_VAR_POP(y , x)}} I guess from here that the 1. alternative may give slightly better runtimes - but not significantly; but in the 2. case the "original" evalutators would do the real work about why do I need to change a bit the existing UDAFs: all these regr_* functions are required to only do any work when neither of {{x}} and {{y}} is null ({{regr_sxx(x,y)}}) which way seems like the better approach?
2. I think you need to create GenericUDAF for each of these? And also you need to support "OVER"? see https://docs.oracle.com/cd/B19306_01/server.102/b14200/functions132.htm
3. [~pxiong] now I see that I never had any chance to not declare these as aggregators :) patch #1) I've retrofitted some existing aggregators to service the regr_ methods.
4. Here are the results of testing the latest attachment: https://issues.apache.org/jira/secure/attachment/12857491/HIVE-15978.1.patch {color:green}SUCCESS:{color} +1 due to 2 test(s) being added or modified. {color:red}ERROR:{color} -1 due to 1 failed/errored test(s), 10420 tests executed *Failed tests:* {noformat} org.apache.hadoop.hive.cli.TestCliDriver.testCliDriver[udaf_binarysetfunctions] (batchId=35) {noformat} Test results: https://builds.apache.org/job/PreCommit-HIVE-Build/4091/testReport Console output: https://builds.apache.org/job/PreCommit-HIVE-Build/4091/console Test logs: http://104.198.109.242/logs/PreCommit-HIVE-Build-4091/ Messages: {noformat} Executing org.apache.hive.ptest.execution.TestCheckPhase Executing org.apache.hive.ptest.execution.PrepPhase Executing org.apache.hive.ptest.execution.ExecutionPhase Executing org.apache.hive.ptest.execution.ReportingPhase Tests exited with: TestsFailedException: 1 tests failed {noformat} This message is automatically generated. ATTACHMENT ID: 12857491 - PreCommit-HIVE-Build
5. [~kgyrtkirk] can you create a RB for this?
6. [~ashutoshc] - i've uploaded the patch to the reviewboard: https://reviews.apache.org/r/57568/
7. #2) use decimal averaging aggregator in avgx/avgy
8. +1 pending tests
9. Here are the results of testing the latest attachment: https://issues.apache.org/jira/secure/attachment/12858807/HIVE-15978.2.patch {color:green}SUCCESS:{color} +1 due to 2 test(s) being added or modified. {color:red}ERROR:{color} -1 due to 1 failed/errored test(s),

10427 tests executed *Failed tests:* {noformat} org.apache.hadoop.hive.cli.TestCliDriver.testCliDriver[udaf_binarysetfunctions] (batchId=35) {noformat} Test results: https://builds.apache.org/job/PreCommit-HIVE-Build/4145/testReport Console output: https://builds.apache.org/job/PreCommit-HIVE-Build/4145/console Test logs: http://104.198.109.242/logs/PreCommit-HIVE-Build-4145/ Messages: {noformat} Executing org.apache.hive.ptest.execution.TestCheckPhase Executing org.apache.hive.ptest.execution.PrepPhase Executing org.apache.hive.ptest.execution.ExecutionPhase Executing org.apache.hive.ptest.execution.ReportingPhase Tests exited with: TestsFailedException: 1 tests failed {noformat} This message is automatically generated. ATTACHMENT ID: 12858807 - PreCommit-HIVE-Build

10. #3) stabilize qtest output by using round() function
11. I will be the HiveQA today! :) Here are the results of testing the latest attachment: https://issues.apache.org/jira/secure/attachment/12858977/HIVE-15978.3.patch {color:green}SUCCESS:{color} +1 due to 2 test(s) being added or modified. {color:red}ERROR:{color} -1 due to 1 failed/errored test(s), 10419 tests executed *Failed tests:* {noformat} org.apache.hadoop.hive.cli.TestSparkCliDriver.org.apache.hadoop.hive.cli.TestSparkCliDriver (batchId=97) {noformat} Test results: https://builds.apache.org/job/PreCommit-HIVE-Build/4171/testReport Console output: https://builds.apache.org/job/PreCommit-HIVE-Build/4171/console Test logs: http://104.198.109.242/logs/PreCommit-HIVE-Build-4171/ Messages: {noformat} Executing org.apache.hive.ptest.execution.TestCheckPhase Executing org.apache.hive.ptest.execution.PrepPhase Executing org.apache.hive.ptest.execution.ExecutionPhase Executing org.apache.hive.ptest.execution.ReportingPhase Tests exited with: TestsFailedException: 1 tests failed {noformat}
12. +1
13. pushed to master, thank you Ashutosh for the review!
14. Doc note: These new functions need to be documented in the wiki. * [Hive Operators and UDFs -- UDAFs | https://cwiki.apache.org/confluence/display/Hive/LanguageManual+UDF#LanguageManualUDF-Built-inAggregateFunctions(UDAF)] Added a TODOC2.2 label.