AtScale Internal Data Types

This table lists the AtScale internal data types and the corresponding external SQL data types that are supported by the data warehouses that work with AtScale.

The AtScale engine maps the SQL type to the AtScale internal type automatically. AtScale uses these internal data types during query planning and optimization. All types are nullable.

If you are using Hadoop as a data warehouse: You can use more than one SQL-on-Hadoop engine, each engine running different types of queries. For example, you might use Hive to build aggregates, and Spark SQL to run your queries. If you use more than one SQL-on-Hadoop engine, make sure that the data types in your tables are supported by both engines.

AtScale Data Type	Description	SQL Data Type/SQL Engine Support
INT	32-bit signed integer (minimum value -2^31, maximum value 2^31-1)	INT (all); SMALLINT, TINYINT (Hive, Impala); INT64 (Google BigQuery)
INTEGER	An arbitrary-precision integer (subject to limits imposed by SQL engine)	BYTE (Spark SQL)
LONG	64-bit signed integer (minimum value -2^63, maximum value 2^63-1)	INT64 (Google BigQuery) BIGINT (Hive, Impala)LONG (Spark SQL)
FLOAT	32-bit floating point (IEEE-754 single precision)	FLOAT (all); REAL (Impala); FLOAT64 (Google BigQuery)
DOUBLE	64-bit floating point (IEEE-754 double precision)	DOUBLE (all) FLOAT64 (Google BigQuery)
DECIMAL(p,s)	An arbitrary-precision decimal (subject to limits imposed by SQL engine)	DECIMAL (all), NUMERIC (Google BigQuery, Redshift, Snowflake)
STRING	Zero or more UTF-8 characters, up to a limit of 2^31-1 characters (a lower limit may be imposed	STRING (all); ARRAY, CHAR, VARCHAR (Hive, Impala);

	by the SQL engine, client or connector)	BINARY, MAP, STRUCT (Hive, Spark SQL); UNIONTYPE (Hive)
BOOLEAN	true or false	BOOLEAN (all)
DATETIME	UNIX-style timestamps	DATETIME (Google BigQuery) TIME (Google BigQuery) TIMESTAMP (all)
DATE	A day in the Gregorian calendar, for example: 2015-06-17 (SQL engine support may be limited)	DATE (Hive, Spark SQL)

Restrictions

The use of arbitrary precision numbers, aka "DECIMAL" types, with AtScale is subject to the restrictions listed below. Additionally, AtScale's DECIMAL support is limited by the capabilities of the connected data warehouse platform. If AtScale Technical Support determines that the underlying data platform has a DECIMAL calculation error, AtScale Inc. will refer you to the appropriate vendor for resolution.

> **SQL Queries Only**: As of 2020.3.1, when a query contains a mathematical expression that combines two Numeric/Decimal measures with different scales or precision values, AtScale delegates the Numeric / Decimal arithmetic behavior to the underlying data source rather than failing the query. If the data warehouse is changed, SQL results may vary slightly depending on the differences of the Numeric / Decimal behavior between the two data warehouses.

- XMLA: XMLA restricts decimal-precision to four digits to the right of the decimal point. As of AtScale 2019.2.2, AtScale can only declare the XMLA decimal(p,s) type for values that meet that requirement. If the value exceeds four digits to the right of the decimal point, AtScale declares the data as an XMLA double type.
- PostgreSQL: AtScale does not support the DECIMAL(p,s) type, aka NUMERIC(p,s), on the PostgreSQL database.
- MAP Columns: AtScale does not support the DECIMAL(p,s) type for MAP type columns.
- Google BigQuery: Only supports 38 digits of precision and 9 fractional digits. It uses data type name NUMERIC for the equivalent of DECIMAL(38,9).
- Hive: In versions of Hive before 2.3, mathematical expressions containing both DECIMAL and floating point data types could produce wrong results. These issues were resolved in Hive 2.3. See the following issues for more information:
  https://issues.apache.org/jira/browse/HIVE-13380
  https://issues.apache.org/jira/browse/HIVE-5848
- Impala: Decimal operations on Impala versions prior to 3.0 (CDH versions prior to 6.0) have the following known platform issues:

- Integer divided by a Decimal can result in incorrect results.
- Decimal multiplied by a Decimal can result in unexpected NULL results.
- Decimal multiplied by an Integer can result in unexpected NULL results.

Additionally, on versions of Impala prior to 3.0, some of the following issues have regressed since their stated fixed version, or require an unsupported query parameter to activate (https://www.cloudera.com/documentation/enterprise/5-14-x/topics/impala_decimal_v2.html):

- DECIMAL division returns an incorrect result: https://issues.apache.org/jira/browse/IMPALA-6429
- DECIMAL multiply overflows early: https://issues.apache.org/jira/browse/IMPALA-4939
- DECIMAL V2 multiply result type: https://issues.apache.org/jira/browse/IMPALA-4940
- DECIMAL divide result type: https://issues.apache.org/jira/browse/IMPALA-4370
- DECIMAL modulo operator is overflowing: https://issues.apache.org/jira/browse/IMPALA-4964
- DECIMAL Error on overflow: https://issues.apache.org/jira/browse/IMPALA-5017
- DECIMAL V2 round when casting to/from DECIMAL, part 2: https://issues.apache.org/jira/browse/IMPALA-5014

Backwards Compatibility

In 2019.1.0, AtScale introduced support for type DECIMAL(p,s), which includes precision and scale. AtScale support is deprecated for DECIMAL without precision and scale.

After upgrading AtScale to 2019.1.0 or later, the cube canvas for pre-existing AtScale cubes notifies you of any DECIMAL types without precision and scale that require conversion. AtScale displays warnings at the bottom of the cube canvas if any physical or calculated columns contain DECIMAL types without precision and scale.

To convert in:

- Physical columns: A warning appears in the cube canvas. Select "Fix It" in the warning to refresh the dataset, which fixes all physical columns in the dataset.
- Calculated columns: A warning appears in the cube canvas for each calculated column using an unsupported DECIMAL type. For each warning, select "Fix It" in the warning to open the calculated column dialog and update the DECIMAL expression with the correct precision and scale. Select "Test" to retest the calculated column, then select "Fix."

After fixing all DECIMAL types, republish the cube.