General Information

Last generated: March 01, 2018

Table of Contents

General Information

This page includes links to various note pages, including our glossary, our trademarks page, and a few other topic pages that you may find useful.

Links to Splice Machine Notes Pages

Link	Description
Splice Machine License	Splice Machine Software End User License Agreement web page.
Spark Overview	A brief overview of Spark.
Using our Documentation	Introduces our documentation conventions and provides navigation and search tips.
Our Documentation Examples	Describes the simple database that we use for many of the examples in our documentation.
Trademarks	A list of trademarks that are referenced in our documentation.
Glossary	Definitions for terms used in our documentation that you might not know.

Splice Machine Editions

This page summarizes the features available in the different editions of Splice Machine.

If you're using the Community Edition of Splice Machine and want to learn more about upgrading to our Enterprise or Cloud editions, please please Contact Splice Machine Sales today.

Feature Comparison

This table summarizes the features that are available in each edition of Splice Machine:

Features	Cloud Edition	Enterprise Edition	Community Edition
On-Demand Compute Nodes and Storage	✓		
Managed Backups and Restores	✓		
Integrated Zeppelin Notebooks	✓		
Splice Machine Cloud Manager	✓		
Scale-Out Architecture	✓	✓	✓
ANSI SQL	✓	✓	✓
Concurrent Acid Transactions	✓	✓	✓
OLAP and OLTP Resource Isolation	✓	✓	✓
Distributed In-Memory Joins, Aggregations, Scans, and Groupings	✓	✓	✓
Cost-Based Statistics / Query Optimizer	✓	✓	✓
Hybrid Row-based and Columnar Storage	✓	✓	✓
Compaction Optimization	✓	✓	✓
Stored Procedures, Triggers, User-Defined Functions	✓	✓	✓
Apache Kafka-enabled Streaming	✓	✓	✓
Virtual Table Interfaces	✓	✓	✓
PL/SQL Support	✓	✓	
Backup and Restore Capabilities	✓	✓	
Column Level Access Control	✓	✓	
Encryption	✓	✓	

Features	Cloud Edition	Enterprise Edition	Community Edition
Security Features, including Kerberos	✓	✓	
LDAP Support		✓	
New Releases and Maintenance Updates	✓	✓	✓

Additional Materials and Support

This table summarizes the additional materials and support that are available for each edition of Splice Machine:

Features	Cloud Edition	Enterprise Edition	Community Edition
Tutorials	✓	✓	✓
Forums	✓	✓	✓
Videos	✓	✓	✓
Online Documentation	✓	✓	✓
Community Support	✓	✓	✓
24/7 Support via Web and Phone	✓	✓	
Complimentary Account Management Services	✓	✓	
GitHub Repository	✓	✓	✓

Licensing

This table compares the pricing policies for the different editions of Splice Machine:

Edition	Pricing Policy
Cloud Edition	On compute and storage units per month basis
Enterprise Edition	On a per node per year basis
Community Edition	Free

The Splice Machine In-Memory Engine

This topic provides an overview of the Splice Machine in-memory engine, which tremendously boosts OLAP (analytical) query performance. Splice Machine use Apache Spark as our in-memory engine and automatically detects and directs OLAP queries to that engine.

This topic presents a very brief overview of Spark terminology and concepts.

> **NOTE:** If you're not yet familiar with Spark, we recommend visiting the Apache Spark web site, spark.apache.org, to learn the basics, and for links to the official Spark documentation.

Spark Overview

Apache Spark is an open source computational engine that manages tasks in a computing cluster. Spark was originally developed at UC Berkeley in 2009, and then open sourced in 2010 as an Apache project. Spark has been engineered from the ground up for performance, exploiting in-memory computing and other optimizations to provide powerful analysis on very large data sets.

Spark provides numerous performance-oriented features, including:

- ability to cache datasets in memory for interactive data analysis
- abstraction
- integration with a host of data sources
- very fast data analysis
- easy to use APIs for operating on large datasets, including numerous operators for transforming and manipulating data, in Java, Scala, Python, and other languages
- numerous high level libraries, including support for machine learning, streaming, and graph processing
- scalability to thousands of nodes

Spark applications consist of a driver program and some number of worker programs running on cluster nodes. The data sets (RDDs) used by the application are distributed across the worker nodes.

Spark Terminology

Splice Machine launches Spark queries on your cluster as Spark *jobs*, each of which consists of some number of *stages*. Each stage then runs a number of *tasks*, each of which is a unit of work that is sent to an *executor*.

The table below contains a brief glossary of the terms you'll see when using the Splice Machine Database Console:

Term	Definition
Action	A function that returns a value to the driver after running a computation on an *RDD*. Examples include `save` and `collect` functions.
Application	A user program built on Spark. Each application consists of a *driver program* and a number of *executors* running on your cluster. An application creates *RDDs*, transforms those RDDs, and runs *actions* on them. These result in a directed acyclic graph (DAG) of operations, which is compiled into a set of *stages*. Each stage consists of a number of *tasks*.
DAG	A **D**irected **A**cyclic **G**raph of the operations to run on an RDD.
Driver program	This is the process that's running the `main()` function of the application and creating the `SparkContext` object, which sends jobs to *executors*.
Executor	A process that is launched (by the driver program) for an *application* on a *worker node*. The executor launches *tasks* and maintains data for them.
Job	A parallel computation consisting of multiple *tasks* that gets spawned in response to a Spark *action*.
Partition	A subset of the elements in an *RDD*. Partitions define the unit of parallelism; Spark processes elements within a partition in sequence and multiple partitions in parallel.
RDD	A **R**esilient **D**istributed **D**ataset. This is the core programming abstraction in Spark, consisting of a fault-tolerant collection of elements that can be operated on in parallel.
Stage	A set of tasks that run in parallel. The stage creates a *task* for each partition in an *RDD*, serializes those tasks, and sends those tasks to *executors*.
Task	The fundamental unit of work in Spark; each task fetches input, executes operations, and generates output.
Transformation	A function that creates a new *RDD* from an existing *RDD*.
Worker node	A cluster node that can run application code.

See Also

- About the Splice Machine Database Console
- User Interface Features of the Splice Machine Database Console
- Managing Queries with the Console

- Using Spark Libraries with Splice Machine
- The Apache Spark web site, spark.apache.org

Using the Splice Machine Documentation

This topic helps orient you to the Splice Machine documentation, in these sections:

- Splice Machine Layout explains the layout of our documentation web and shows you how to use the various navigation tools to land on the pages in which you're interested.
- Navigating our Documentation summarizes and links to all of the top-level sections in the Splice Machine documentation suite.

Splice Machine Layout

Here's an image of the top portion of our documentation screen, which is called the *topbar*:

The topbar features these elements:

- Click the *Splice Machine Documentation* title to return you to the home page of our documentation.
- Click the Navigation Toggle to toggle the sidebar off (to expand the width of the main content) or off.
- The top navigation menus includes our three main menus, which link to the main topic sections.
- A sidebar for navigating to topics within the main sections.
- A search bar that you can use to search for topic titles within the documentation.
- The main content area, which contains the content of each topic.

Internal and External Links

Links to other pages in the documenation are shown in underlined blue.

Links to external sites are also shown in underlined blue, and include a boxed arrow symbol that indicates the link will automatically open in a separate browser tab or window. For example, this link opens the Splice Machine web home page in a separate browser tab.

Navigating our Documentation

This site includes documentation for both of our Splice Machine products. The topic sections are organized into three main categories, each of which has detailed sections and is represented by one of the menus at the top of each page (*Splice Machine, DB-Service Only, On-Premise-DB Only*). The sidebar navigation (on the left) automatically changes whenever you select a new section of the docs.

The following table summarizes the main sections of our documentation:

Menu	Section	Description
Splice Machine	Welcome	Information about our database: the basis of all Splice Machine products.
	Tutorials	A collection of tutorials that walk you through numerous specific tasks to help you quickly learn how to use your Splice Machine database more productively.
	SQL Reference Manual	The reference manual for the Splice Machine implementation of SQL.
	Developer's Manual	Topics of interest to all developers working with Splice Machine.
	Command Line Reference	The reference manual for our Splice Machine command line interface.
	Database Console Guide	An introduction to the Splice Machine Spark Database Console.
	Release Notes	Information about new features, improvements, and fixes in the current database release.
General Information	General information about our database products and our documentation.	
DB-Service Only	Welcome	The content in this section is specific to our cloud-managed database service product.
	Cloud Manager Guide	A guide to our Cloud Manager, which is the Dashboard from which you create, manage, and use your clusters.
	Using Zeppelin	A guide to using Zeppelin notebooks to work with your Splice Machine databases.
	Release Information	Release notes, workarounds, and other information about this release.
On-Premise-DB Only	Welcome	The content in this section is specific to our on-premise database product.
	Installing Splice Machine	Step-by-step instructions for installing the on-premise version of Splice Machine on compatible platforms.
	On-Premise Administration	Topics that describe the administrative tasks associated with installing, configuring, and maintaining your on-premise Splice Machine database.

Menu	Section	Description
	Best Practices and Troubleshooting	Tips for best practices and solving common problems.

About Our Documentation Examples Database

This topic describes the database that we have created to provide code examples throughout our documentation suite. We've pulled in basic seasonal statistics for two Major League Baseball teams, though all names have been changed. We're calling this our `DocsExamplesDb` database.

Our `DocsExamplesDb` database features these tables:

Table	Contains
Players	The player's name, ID, and other general information.
Salaries	The salary for each player ID for each season.
Batting	Batting statistics, per season, for each player ID.
Fielding	Fielding statistics, per season, for each player ID.
Pitching	Pitching statistics, per season, for each pitcher's player ID.

The tables were populated with data found on the Internet, primarily from the baseball-reference.com site.

Table Schemas

Our example tables are all stored in a schema named `SPLICEBBALL`. This section describes the fields in each of our `DocsExamplesDb` tables.

The Players Table

The `SPLICEBBALL.Players` table contains these columns:

Column Name	Type	Description
`ID`	`SMALLINT`	The unique player ID, assigned upon insertion.
`Team`	`VARCHAR`	The abbreviated name of the player's team.
`DisplayName`	`VARCHAR`	The name we use when displaying this player.
`Position`	`CHAR(2)`	The abbreviation for the player's main position, e.g. P, C, OF, 1B.
`Birthdate`	`DATE`	The birth date of the player.

The Salaries Table

The `SPLICEBBALL.Salaries` table contains these columns:

Column Name	Type	Description
`ID`	`SMALLINT`	The unique player ID.
`Season`	`SMALLINT`	The season (year).
`Salary`	`BIGINT`	The player's salary for the season.

The Batting Table

The `SPLICEBBALL.Batting` contains these columns:

Column Name	Type	Description
`ID`	`SMALLINT`	The unique player ID.
`Season`	`SMALLINT`	The season (year).
`Games`	`SMALLINT`	The number of games in which the player batted.
`PlateAppearances`	`SMALLINT`	The number of times the player made a plate appearance.
`AtBats`	`SMALLINT`	The number of official at bats.
`Runs`	`SMALLINT`	The number of runs scores.
`Hits`	`SMALLINT`	How many hits by the player.
`Singles`	`SMALLINT`	How many singles hit by the player. This value is computed by a triggered function.
`Doubles`	`SMALLINT`	How many doubles hit by the player.
`Triples`	`SMALLINT`	How many triples hit by the player.
`HomeRuns`	`SMALLINT`	How many home runs hit by the player.
`RBI`	`SMALLINT`	How many Runs Batted In by the player.
`StolenBases`	`SMALLINT`	How many bases the player stole.

Column Name	Type	Description
CaughtStealing	SMALLINT	How many times the player was caught attempting to steal a base.
Walks	SMALLINT	The number of walks the player drew.
Strikeouts	SMALLINT	The number of times the player walked.
DoublePlays	SMALLINT	How many times the player hit into a double play.
HitByPitches	SMALLINT	The number of times the player was hit by a pitch.
SacrificeHits	SMALLINT	The number of sacrifice bunts the player hit.
SacrificeFlies	SMALLINT	The number of sacrifice flies the player hit.
IntentionalWalks	SMALLINT	The number of intentional walks issued to the player.
Average	DECIMAL	The player's batting average. This value is computed by a triggered function.
TotalBases	SMALLINT	The total number of bases for the player. This value is computed by a triggered function.
OnBasePercentage	DECIMAL	The percentage of times the player reached base. This value is computed by a triggered function.
Slugging	DECIMAL	The slugging average of the player. This value is computed by a triggered function.
OnBasePlusSlugging	DECIMAL	The OPS for the player. This value is computed by a triggered function.

The Fielding Table

The `SPLICEBBALL.Fielding` table contains these columns:

Column Name	Type	Description
ID	SMALLINT	The unique player ID.
Season	SMALLINT	The season (year).
FldGames	SMALLINT	How many games the player was in the field for.
Chances	SMALLINT	The number of fielding chances the player had.
Putouts	SMALLINT	The number of putouts the player had.
Assists	SMALLINT	The number of assists the player had.
Errors	SMALLINT	The number of errors committed by the player.
FldDoublePlays	SMALLINT	The number of doubles plays in which the player was involved in as a fielder.
Percentage	DECIMAL	The percentage of opportunities for outs that the player successfully completed.
TZAboveAverage	SMALLINT	A fielding metric: total zone runs above average for his position.
TZAboveAveragePer1200	SMALLINT	Total zone runs extrapolated for 1200 innings.
RunsSaved	SMALLINT	The number of runs saved in the field by the player.
RunsSavedAboveAvg	SMALLINT	The number of runs saved by the player over the average number saved for his position.
RangeFactorPerNine	DECIMAL	A fielding metric that evaluates the average number of putouts and assists per nine innings,
RangeFactorPerGame	DECIMAL	A fielding metric that evaluates the average number of putouts and assists per game played,
PassedBalls	SMALLINT	The number of passed balls for catchers.
WildPitches	SMALLINT	The number of wild pitches for pitchers.
FldStolenBases	SMALLINT	The number of stolen bases given up by a pitcher or catcher.
FldCaughtStealing	SMALLINT	The number of players caught stealing by a pitcher or catcher.

Column Name	Type	Description
FldCaughtStealingPercent	DECIMAL	For catchers and pitchers, the percentage of attempted stolen bases that were successful.
FldLeagueCaughtStealingPercent	DECIMAL	For pitchers and catchers, the league average percentage of attempted stolen bases that were successful.
Pickoffs	SMALLINT	For pitchers and catchers, the number of runners picked off.
FldInnings	DECIMAL	The number of innings in which the player was in the field.

The Pitching Table

The `SPLICEBBALL.Pitching` table contains these fields:

Column Name	Type	Description
ID	SMALLINT	The unique player ID.
Season	SMALLINT	The season (year).
Wins	SMALLINT	How many games the pitcher won.
Losses	SMALLINT	How many games the pitcher lost.
Games	SMALLINT	The number of games in which the pitcher appeared.
GamesStarted	SMALLINT	The number of games the pitcher started.
GamesFinished	SMALLINT	The number of games the pitcher finished.
CompleteGames	SMALLINT	The number of complete games by the pitcher.
Shutouts	SMALLINT	The number of shutout games thrown by the pitcher.
Saves	SMALLINT	The number of games saved by the pitcher.
Innings	DECIMAL	The number of innings pitched.
Hits	SMALLINT	The number of hits given up by the pitcher.
Runs	SMALLINT	The number of runs give up by the pitcher.

Column Name	Type	Description
EarnedRuns	SMALLINT	The number of earned runs give up by the pitcher.
HomeRuns	SMALLINT	How many homeruns the pitcher gave up.
Walks	SMALLINT	How many walks the pitcher issued.
IntentionalWalks	SMALLINT	How many intentional walks the pitcher issued.
Strikeouts	SMALLINT	How many batters the pitchers struck out.
HitBatters	SMALLINT	How many batters the pitcher hit with a pitch.
Balks	SMALLINT	How many balks the pitcher committed.
WildPitches	SMALLINT	How many wild pitches were thrown by the pitcher.
BattersFaced	SMALLINT	The number of batters faced by the pitcher.
FieldingIndependent	DECIMAL	A metric (FIP) for pitchers that determines the quality of a pitcher's performance by eliminating plate appearance outcomes that involve defensive play.
ERA	DECIMAL	The pitcher's earned run average. This value is computed by a triggered function.
WHIP	DECIMAL	The number of walks and hits per inning pitched by the player. This value is computed by a triggered function.
HitsPerNine	DECIMAL	The number of hits per nine innings pitched by the player. This value is computed by a triggered function.
HomeRunsPerNine	DECIMAL	The number of home runs per nine innings pitched by the player. This value is computed by a triggered function.
WalksPerNine	DECIMAL	The number of walks per nine innings pitched by the player. This value is computed by a triggered function.

Column Name	Type	Description
`StrikeoutsPerNine`	`DECIMAL`	The number of strikeouts per nine innings pitched by the player. This value is computed by a triggered function.
`StrikeoutsToWalks`	`DECIMAL`	The ratio of strikeouts to walks thrown by the pitcher. This value is computed by a triggered function.

Trademarks

The following table lists third-part trademark information for products mentioned in this documentation suite:

Trademark Holder	Trademarks
Amazon Web Services, Inc.	AWS and Amazon Elastic Compute Cloud are trademarks of Amazon Web Services, Inc.
Apache Software Foundation	Apache, Apache Derby, Apache Spark, HBase, and Hive are trademarks of the Apache Software Foundation. Hadoop is a registered trademark of the Apache Software Foundation.
Apple Computer, Inc.	Mac OS and OS X are registered trademarks of Apple Computer, Inc.
Canonical Limited	Ubuntu is a registered trademark of Canonical Limited.
Cloudera, Inc.	Cloudera is a trademark of Cloudera Corporation.
DbVis Software AB	DbVisualizer is trademark of DbVis Software AB.
EasySoft Limited	EasySoft is a trademark of EasySoft Limited.
Hortonworks, Inc.	Hortonworks and HDP are registered trademarks of Hortonworks, Inc.
Linus Torvalds	Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.
MapR Technologies, Inc.	MapR is a registered trademark of MapR Technologies, Inc.
Microsoft Corporation	Windows is a registered trademark of Microsoft Corporation.
MongoDB, Inc.	MongoDb is a registered trademark of MongoDB, Inc.
Oracle Corporation	MySQL is a trademark of Oracle Corporation or its affiliates. Oracle, Java, and JDBC are registered trademarks of Oracle Corporation or its affiliates.
Pentaho, Inc.	Pentaho is a registered trademark of Pentaho, Inc.
Red Hat, Inc.	Red Hat, Red Hat Linux, Red Hat Enterprise Linux, and CentOS are registered trademarks of Red Hat, Inc
Tableau Software, Inc.	Tableau and Tableau Software are registered trademarks of Tableau Software.
The Open Group	UNIX is a registered trademark of The Open Group in the U.S. and other countries.

Glossary

Term	Definition
ACID Transactions	ACID (**A**tomicity, **C**onsistency, **I**solation, **D**urability) is a set of properties that guarantee that database transactions are processed reliably. In the context of databases, a single logical operation on the data is called a transaction. » *Atomicity* means that if one part of the transaction fails, the entire transaction fails. » *Consistency* ensures that any transaction will bring the database from one valid state to another, which means that any data written to the database must be valid according all rules defined in the database. » *Isolation* ensures that the concurrent execution of transactions results in a system state that would be obtained if transactions were executed in serial order. » *Durability* means that once a transaction has been committed, it will remain so, even in the event of power loss, crashes, or errors.
Auto-sharding	The database is automatically and transparently partitioned (sharded) across low cost commodity nodes, allowing scale-out of read and write queries, without requiring changes to the application.
BI Tools	**B**usiness **I**ntelligence Tools
CDH	Clouderas ***C**loudera **D**istribution Including Apache Hadoop*, a popular Hadoop platform.
Column-Oriented Data Model	A model for storing data in a database as sections of columns, rather than as rows of data. In a column-oriented database, all of the values in a column are serialized together.
Concurrency	The ability for multiple users to access data at the same time.
CRM	**C**ustomer **R**elationship **M**anagement
Cross-table, cross-row transactions	A transaction (a group of SQL statements) can modify multiple rows (cross-row) in multiple tables (across tables).
CRUD	**C**reate, **R**ead, **U**pdate, **D**elete. The four basic functions of persistent storage.
DAG	**D**irected **A**cyclic **G**raph. A directed graph with no directed cycles, meaning that no path through the graph loops back to its starting point. DAGs are used for various computational purposes, including query optimization in some databases.
Database Statistics	A form of dynamic metadata that assists the query optimizer in making better decisions by tracking distribution of values in indexes and/or columns.

Term	Definition
Database Transaction	A sequence of database operations performed as a single logical unit of work.
ERP	**E**nterprise **R**esource **P**lanning is business management software that a company can use to collect, store, manage and interpret data from many business activities.
Foreign Key	A column or columns in one table that references a column (typically the primary key column) of another table. Foreign keys are used to enture referential integrity.
Full join support	Databases use join operations to combine fields from multiple tables by using values common to each. Full join support means that the Database Management System support all five ANSI-standard types of join operations: Inner join, left outer join, right outer join, full outer join, and cross join.
Hadoop	An Apache open source software project that enables the distributed processing of large data sets across clusters of commodity servers. It is designed to scale up from a single server to thousands of machines, with a very high degree of fault tolerance.
HBase	A column-oriented database management system that is part of the Apache Hadoop framework and runs on top of HDFS.
HCatalog	Apache HCatalog is a table and storage management layer for Hadoop that enables users with different data processing tools to more easily read and write data on the grid.
HDFS	**H**adoop **D**istributed **F**ile **S**ystem. A distributed file system that stores data on commodity hardware and is part of the Apache Hadoop framework. It links together the file systems on many local nodes to make them into one big file system.
HDP	***H**ortonworks **D**ata **P**latform* includes Apache Hadoop and is used for storing, processing, and analyzing large volumes of data. The platform is designed to deal with data from many sources and formats.
HIVE	Apache Hive is a data warehouse infrastructure built on top of Hadoop for providing data summarization, query, and analysis.
HR	**H**uman **R**esources.
JDBC	***J**ava **D**ata**B**ase **C**onnection*. An API specification for connecting with databases using programs written in Java.
JSON	An open standard format that uses human-readable text to transmit data objects consisting of attributevalue pairs. It is used primarily to transmit data between a server and web applications, as an alternative to XML.
JVM	**J**ava **V**irtual **M**achine. The code execution component of the Java platform.
Key-Value Data Model	A fundamental and open-ended data model that allows for extension without modifying existing data. Data is represented in pairs: name (or key) and a value that is associated with that name. Also known as key-value pair, name-value pair, field-value pair, and attribute-value pair.

Term	Definition
Map Reduce	MapReduce is a programming model and an associated implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster. MapReduce programs typically include these steps: 1. Each worker node applies the `Map()` function to filter and sort local data. 2. Worker nodes redistribute (shuffle) data based on output keys produced by the `Map()` step, so that all data belonging to one key is located on the same node. 3. Worker nodes process each group of output data in parallel to produce results. 4. The MapReduce system collects the results and sorts them to produce the final outcome.
MapR	*MapR* is a complete distribution for Apache Hadoop that packages more than a dozen projects from the Hadoop ecosystem to provide a broad set of big data capabilities.
Multi-partition transactions	A database, like Splice Machine, that allows transactions for a table distributed as multiple partitions across multiple nodes in a cluster.
MVCC	**M**ultiVersion **C**oncurrency **C**ontrol is a method used to control concurrent access to a database. Concurrency control is needed to bypass the potential problem of someone viewing (reading) a data value while another is writing to the same value.
MySQL	An open source Relational Database Management System (RDBMS) that uses Structured Query Language (SQL).
NewSQL	NewSQL is a class of modern relational database management systems that seek to provide the same scalable performance of NoSQL systems for online transaction processing (OLTP) read-write workloads while still maintaining the ACID guarantees of a traditional database system.
NoSQL	A NoSQL database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases.
ODBC	***O**pen **D**ata**B**ase **C**onnectivity*. An open standard API for accessing database management systems, designed to be independent of any specific database or operating system.
OLAP	**O**n**L**ine **A**nalytical **P**rocessing. An approach to quickly answering multi-dimensional analytical queries in the Business Intelligence world. OLAP tools allow users to analyze multidimensional data interactive from multiple perspectives.
OLTP	**O**n**L**ine **T**ransaction **P**rocessing. A class of information processing systems that facilitate and manage transaction-oriented applications.
Query Optimizer	A critical database management system (DBMS) component that analyzes SQL queries and determines efficient execution mechanisms, known as query plans. The optimizer typically generates several plans and then selects the most efficient plan to run the query.

Term	Definition
Referential Integrity	A property of data that requires every value of one column in a table to exist as a value of another column in a different (or the same) table. This term is generally used to describe the function of foreign keys.
Relational Data Model	The model, developed by E.F. Codd, upon which relational database are based. Relational tables have these properties: » Data is presented as a collection of relations » Each relation is depicted as a table » Columns are attributes that belong to the entity modeled by the table » Each row represents a single entity (a record) » Every table has a set of attributes, a key, that unique identifies each entity
REST	**RE**presentational **S**tate **T**ransfer. A simple, stateless, client-server protocol use for networked applications, which uses the HTTP requests to communicate among machines. RESTful applications use HTTP requests to post (create or update) data, to read data, and to delete data. Collectively, these are known as CRUD operations.
Rollback	An operation that returns the database to some previous state, typically used for recovering from database server crashes: by rolling back any transaction that was active at the time of the crash, the database is restored to a consistent state.
Scale Out	A database architecture that doesnt rely on a single controller and scales by adding processing power coupled with additional storage.
Scale Up	An architecture that uses a fixed controller resource for all processing. Scaling capacity happens by adding processing power to the controller or (eventually) upgrading to a new (and expensive) controller.
Sharding	Horizontal partitioning in a database: that the data is split among multiple machines while ensuring that the data is always accessed from the correct place. See Auto-sharding.
Spark	Apache Spark is an open-source cluster computing framework that uses in-memory primitives to reduce storage access and boost database performance. Spark allows user applications to load data into a cluster's memory and repeatedly query that data.
Trigger	A database trigger is code that is run automatically in response to specific events on specific tables or views in your database. Triggers are typically configured to maintain data integrity, such as ensuring that an updated value is in range.
YARN	***Y**et **A**nother **R**esource **N**egotiator*. YARN assigns CPU, memory, and storage to applications running on a Hadoop cluster, and enables application frameworks other than MapReduce (like Spark) to run on Hadoop.

Term	Definition
ZooKeeper	Part of the Apache Hadoop framework, ZooKeeper provides a centralized infrastructure and services that enable synchronization across a cluster. ZooKeeper maintains common objects needed in large cluster environments. Examples of these objects include configuration information, hierarchical naming space, and so on. Applications can leverage these services to coordinate distributed processing across large clusters.