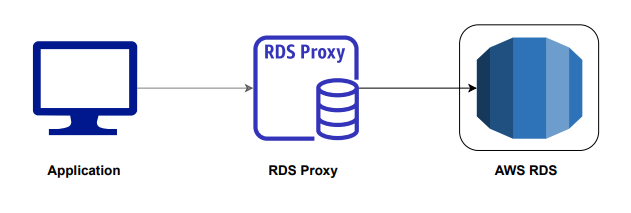
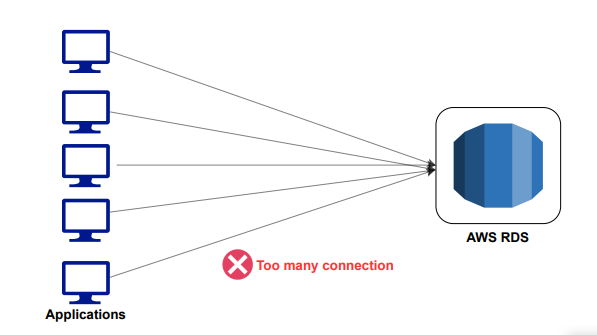
**What is RDS Proxy?**



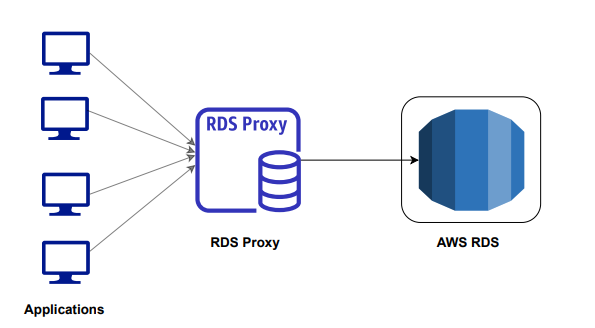
* AWS RDS proxy is a fully managed database that serves as a proxy layer between the application and the database.
* Client/application is pointed to RDS proxy endpoint, RDS Proxy sits between the database and RDS to manage the DB connection.
* RDS Proxy works by pooling and sharing DB connections and thus makes applications more scalable as well as resilient to database failures
* If you are running applications with unpredictable workloads, which require frequently open and close database connections, and you want to maintain high availability during transient database failure, the Amazon RDS proxy is the solution

**Challenges in connecting to RDS**



* Many applications can create more connections to the databases which exhausts CPU and memory and also too many connections.
* For each database connection, the DB server consumes memory and CPU that cannot allocate enough resources for query execution.
* AWS Lambda-based serverless applications make thousands of database connections in a short period of time.

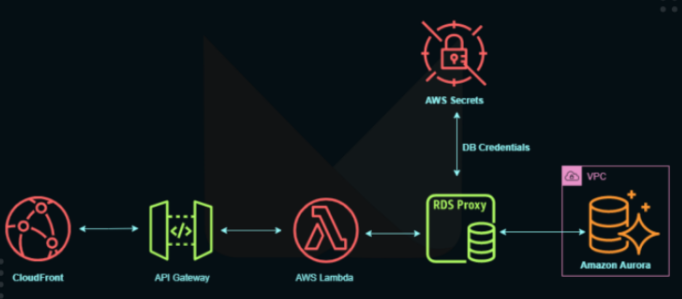
**How does it work?**



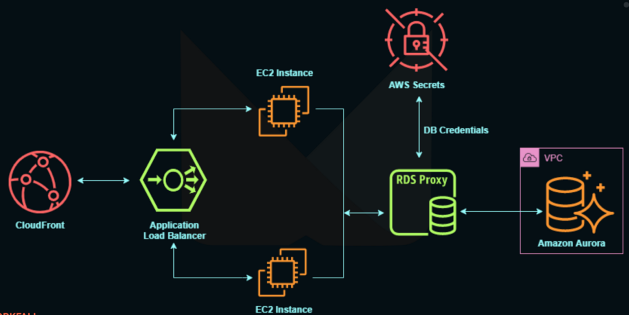
* RDS proxy works by pooling and sharing DB connections that enable applications to share and re-use database connections thus reducing the load on the database and improve application performance.
* We are no longer need code that handles cleaning up idle connections and managing connection pools.
* Opening and closing a new database connection is CPU-intensive whereas additional memory is needed for each open connection.
* Each database transaction uses one underlying database connection which can be reused once the transaction has finished. This transaction-level reuse is called **connection** **multiplexing** (or connection reuse).
* In **connection multiplexing**, database connections are shared between client connections which helps minimize the resource overhead on the database server.
* RDS proxy helps improve application availability during failure scenarios such as a database failover.
* RDS Proxy gives you the choice to use IAM authentication for connecting to the database, thus removing the need for database credentials in the application code.
* RDS proxy helps to reduce the failover times for Aurora and RDS DB instances by 66%.
* Database access can be controlled through integration with AWS Secrets Manager and IAM Role.
* It is compatible with various database engines like Aurora MySQL, RDS MySQL, Aurora DB Cluster, PostgreSQL.
* DB credentials are managed by AWS Secrets Manager and are not hard-coded into the application for improved security

**Use Cases of RDS Proxy**

* RDS proxy can be used for the applications hosted using AWS Lambda (Serverless)



* RDS proxy can be used for the applications hosted on EC2 Instances.



* Unpredictable workloads
* Frequently open and close database connections
* Higher availability during transient database failures

**What are the limitations?**

* RDS proxy must be in the same VPC as the database instance. The proxy cannot be publicly accessible even if the database instance is.
* RDS proxy cannot be used with a self-managed EC2-instance based database.
* RDS proxy cannot be used for Aurora Serverless yet.
* A proxy can only be associated with 1 Database instance

**Pricing:**

* Amazon RDS Proxy is priced per vCPU per hour for each database instance for which it is enabled.
* The price depends on the RDS instance type used by your database. The larger the database instance, the more you end up paying.
* Partial hours are billed in one-second increments with a 10-minute minimum charge.