**Cost Analysis for MySQL RDS on AWS**

To conduct a cost analysis, we need to consider the different factors that contribute to the cost of running a MySQL RDS instance on AWS. The primary components include:

1. **Instance Class and Pricing:**
   * **Instance Type:** db.t3.micro
   * **vCPUs:** 2
   * **RAM:** 1 GiB
   * **Pricing:** $0.020 per hour
2. **Storage Costs:**
   * **Allocated Storage:** 20 GiB
   * **Pricing:** $0.115 per GiB-month
3. **Backup Storage:**
   * **Included Backup Storage:** Up to the size of the database is free
   * **Additional Backup Storage:** $0.095 per GiB-month (if exceeds the database size)
4. **Data Transfer Costs:**
   * **Data Transfer Out to Internet:** $0.09 per GiB (first 1 GB per month is free)

**Monthly Cost Calculation**

**Instance Costs**

* **Instance Hourly Cost:**
  + $0.020 per hour
* **Monthly Instance Cost:**
  + $0.020 \* 24 hours/day \* 30 days/month = $14.40 per month

**Storage Costs**

* **Monthly Storage Cost:**
  + 20 GiB \* $0.115 per GiB-month = $2.30 per month

**Backup Costs**

* Assuming the backup storage used is within the free tier (up to the size of the database), we won't include additional backup costs.

**Data Transfer Costs**

* Assuming minimal data transfer out to the internet beyond the free 1 GB per month, we won't include additional data transfer costs.

**Total Monthly Cost**

**Total Estimated Monthly Cost:**

* Instance Cost: $14.40
* Storage Cost: $2.30
* **Total:** $16.70 per month

**Documentation for the Database**

Here’s a basic structure for documenting your database:

**1. Database Overview**

**Database Name:** auth\_bill\_pro

**Purpose:** The database manages activities, authorizations, participants, and links for a small business application.

**2. Table Descriptions**

**Table: participants**

* **Description:** Stores participant information.
* **Columns:**
  + participant\_id (INT, Primary Key)
  + email (VARCHAR(255))
  + first\_name (VARCHAR(255))
  + last\_name (VARCHAR(255))
  + phone (VARCHAR(20))
  + registration\_date (DATETIME)

**Table: activities**

* **Description:** Logs activities related to participants.
* **Columns:**
  + activity\_id (INT, Primary Key)
  + activity\_date (DATE)
  + actType\_name (INT)
  + participant\_id (INT)
  + actCase\_notes (TEXT)
  + actBillable\_hours (INT)
  + auth\_number (VARCHAR(20))

**Table: auth**

* **Description:** Stores authorization details.
* **Columns:**
  + auth\_number (VARCHAR(20), Primary Key)
  + auth\_begin\_date (DATETIME)
  + auth\_end\_date (DATETIME)
  + auth\_details (TEXT)
  + auth\_rate (DECIMAL(10, 2))
  + auth\_billable\_hours (INT)
  + auth\_remaining\_billable\_hours (INT)
  + participant\_id (INT)

**Table: links**

* **Description:** Stores links associated with participants.
* **Columns:**
  + link\_id (INT, Primary Key)
  + link (VARCHAR(255))
  + participant\_id (INT)

**Table: activity\_types**

* **Description:** Defines the types of activities.
* **Columns:**
  + type\_id (INT, Primary Key)
  + type\_name (VARCHAR(255))
  + type\_desc (TEXT)

**3. Relationships**

* participants are linked to activities through participant\_id.
* auth records are linked to participants through participant\_id.
* links are associated with participants through participant\_id.
* activities are associated with activity\_types through actType\_name.

**4. Key Queries**

**Query 1: Total Billable Hours by Participant**

SELECT p.first\_name, p.last\_name, SUM(a.actBillable\_hours) AS TotalBillableHours

FROM participants p

JOIN activities a ON p.participant\_id = a.participant\_id

GROUP BY p.participant\_id;

**Query 2: Authorization Details**

SELECT auth\_number, auth\_begin\_date, auth\_end\_date, auth\_rate, auth\_billable\_hours, auth\_remaining\_billable\_hours

FROM auth;

**Query 3: Activities and Types**

SELECT a.activity\_id, a.activity\_date, at.type\_name, a.actCase\_notes, a.actBillable\_hours, p.first\_name, p.last\_name

FROM activities a

JOIN participants p ON a.participant\_id = p.participant\_id

JOIN activity\_types at ON a.actType\_name = at.type\_id;

**5. Modules and Macros**

**Module:**

* **Name:** [ModuleName] (replace with actual name)
* **Description:** This module contains VBA code to handle [specific tasks such as data validation, custom calculations, etc.].
* **Key Functions:**
  + Function1(): [Description of what Function1 does]
  + Function2(): [Description of what Function2 does]

**Macro:**

* **Name:** [MacroName] (replace with actual name)
* **Description:** This macro automates [specific tasks such as data import/export, report generation, etc.].
* **Steps:**
  + Step 1: [Description of the first step]
  + Step 2: [Description of the second step]

**6. Maintenance and Backup Procedures**

**Backup Procedures:**

* Automated backups configured on AWS RDS with a retention period of 7 days.
* Manual snapshot taken before any major updates.

**Maintenance Tasks:**

* Regular analysis and optimization of tables.
* Monitoring performance metrics using AWS CloudWatch.
* Regular review of security group settings to ensure secure access.

**7. Migration Steps and Required Software**

**Software Required:**

* Microsoft Access
* MySQL Workbench
* MySQL ODBC Driver
* AWS Account

**Migration Steps:**

1. **Set Up MySQL RDS on AWS:**
   * Create an RDS instance, configure settings, and launch it.
2. **Install MySQL ODBC Driver:**
   * Download and install the appropriate version.
3. **Configure ODBC Data Source:**
   * Add and configure a new data source.
4. **Export Data from Access to Text Files:**
   * Export each table to a text file.
5. **Import Data into MySQL Using MySQL Workbench:**
   * Create tables and use LOAD DATA INFILE to import data.
6. **Link Access Front-End to MySQL RDS:**
   * Use the linked table manager in Access to link tables to the MySQL RDS instance.

**8. RDS Database Configuration**

**Instance Specifications:**

* **Instance Type:** db.t3.micro
* **vCPUs:** 2
* **RAM:** 1 GiB
* **Storage:** 20 GiB (General Purpose SSD)

**Database Engine:**

* **Engine Type:** MySQL
* **Engine Version:** 8.0.35

**Instance Settings:**

* **Instance Identifier:** auth-bill-pro
* **Master Username:** admin
* **Endpoint:** auth-bill-pro.c3q6066gukuc.us-east-2.rds.amazonaws.com
* **Port:** 3306

**Security:**

* **VPC:** Default VPC
* **Security Group:** Custom security group allowing inbound traffic on port 3306 from trusted IP addresses.
* **Encryption:** Enabled for data at rest using AWS-managed keys.

**Backup and Maintenance:**

* **Automated Backups:** Enabled with a retention period of 7 days.
* **Backup Window:** Configured to occur during low-traffic periods.
* **Maintenance Window:** Weekly maintenance window scheduled during low-traffic periods.

**Monitoring and Performance:**

* **CloudWatch Monitoring:** Enabled for performance metrics.
* **Enhanced Monitoring:** Enabled for real-time instance metrics.
* **Parameter Group:** Custom parameter group configured for optimal performance.

**RDS Database Inbound and Outbound Configurations**

**Inbound Configuration**

**Security Group Inbound Rules:**

* **Type:** MySQL/Aurora
* **Protocol:** TCP
* **Port Range:** 3306
* **Source:** Specific IP addresses or CIDR blocks that are allowed to connect to the database (e.g., your office IP, developer IP addresses)

Example:

* **Type:** MySQL/Aurora
* **Protocol:** TCP
* **Port Range:** 3306
* **Source:** 203.0.113.0/24 (replace with your specific IP range)

**Outbound Configuration**

**Security Group Outbound Rules:**

* **Type:** All traffic
* **Protocol:** All
* **Port Range:** All
* **Destination:** 0.0.0.0/0 (allowing all outbound traffic, which is the default setting for most security groups)

Example:

* **Type:** All traffic
* **Protocol:** All
* **Port Range:** All
* **Destination:** 0.0.0.0/0

**Summary of Inbound and Outbound Rules**

**Inbound Rules**

1. **MySQL/Aurora:**
   * **Protocol:** TCP
   * **Port Range:** 3306
   * **Source:** (Your specific IP range or address, e.g., 203.0.113.0/24)

**Outbound Rules**

1. **All Traffic:**
   * **Protocol:** All
   * **Port Range:** All
   * **Destination:** 0.0.0.0/0

These configurations ensure that only trusted sources can access your MySQL RDS instance, while allowing the instance to communicate with external services as needed. If you need more restrictive outbound rules, you can specify specific destinations instead of allowing all traffic.