REVISTA FEDSIDUMSA EN ACCIÓN
Órgano de Difusión Docente - UMSA
Vol. 1, Nro. 1 (2026)

IMPLEMENTACIÓN DE SERVIDOR MCP CON DATA API BUILDER PARA ACCESO DE AGENTES IA EN AZURE

Henry Mamani Chavez [1] Carla Vanesa Mamani Chavez [2]

1. Carrera de Informática, Universidad Mayor de San Andrés, La Paz, Bolivia
hmamani16@umsa.bo | ORCID: 0009-0003-9031-5339
2. Data Engineer Independiente
ORCID: 0000-0002-1556-1578

RESUMEN

La presente investigación tuvo como objetivo implementar y evaluar un servidor basado en el Protocolo de Contexto de Modelo que permita a agentes de inteligencia artificial acceder de manera segura y controlada a bases de datos transaccionales, evitando los riesgos asociados a la generación directa de consultas mediante lenguaje natural. La metodología empleada fue de tipo experimental y consistió en el diseño de una solución utilizando un esquema relacional administrado en un entorno de base de datos transaccional, expuesto mediante una capa de acceso gobernada que publica las entidades como herramientas consumibles por agentes de inteligencia artificial. La población objetivo estuvo conformada por profesionales del sector bancario sin conocimientos de programación, entre ellos contadores, administradores y gerentes, quienes participaron en pruebas controladas orientadas a validar la consulta y recuperación de información de negocio. Los resultados evidenciaron una elevada tasa de resolución correcta de las solicitudes, tiempos de respuesta reducidos y un acceso eficiente a la información operativa. Asimismo, se comprobó la efectividad de los mecanismos de seguridad implementados, observándose la ausencia de vulnerabilidades relacionadas con inyección de código y una adecuada aplicación de controles de permisos sobre los datos consultados. Se concluye que la implementación de un servidor basado en este protocolo constituye una alternativa viable para democratizar el acceso a la información empresarial, al desacoplar los agentes de inteligencia artificial de la lógica de acceso a datos, fortalecer la gobernanza de la información y reducir significativamente la superficie de ataque en comparación con enfoques tradicionales basados en la generación dinámica de consultas.

Palabras clave: *Bases de Datos, Inteligencia Artificial, Computación en la Nube, Seguridad de Datos.*

1. INTRODUCCIÓN

En el entorno empresarial contemporáneo, el acceso oportuno y preciso a la información almacenada en bases de datos transaccionales constituye un elemento fundamental para la toma de decisiones operativas y estratégicas. Sin embargo, gran parte de los usuarios que requieren consultar información corporativa, como contadores, administradores, gerentes y profesionales del área jurídica, carecen de conocimientos técnicos en lenguaje de consulta estructurado, lo que dificulta su interacción directa con los sistemas de gestión de datos. Esta limitación genera dependencia de especialistas en tecnologías de información para la elaboración de consultas y reportes, incrementando los tiempos de respuesta y reduciendo la eficiencia organizacional. El avance de los modelos de lenguaje de gran escala ha permitido la aparición de agentes de inteligencia artificial capaces de interpretar solicitudes en lenguaje natural y transformarlas en consultas a fuentes de datos empresariales. Una de las estrategias más utilizadas consiste en la traducción automática de instrucciones en lenguaje natural a sentencias de lenguaje de consulta estructurado. No obstante, este enfoque presenta importantes desafíos relacionados con la seguridad y la gobernanza de la información, debido a la posibilidad de generar consultas no autorizadas, exponer la estructura interna de la base de datos o introducir vulnerabilidades derivadas de una validación insuficiente de las entradas del usuario (Microsoft, 2025). Con el propósito de mitigar estos riesgos, han surgido arquitecturas orientadas a desacoplar la capa de inteligencia artificial de la capa de acceso a datos. Entre ellas destaca el Protocolo de Contexto de Modelo, un estándar abierto diseñado para permitir que los agentes de inteligencia artificial interactúen con herramientas y sistemas externos de forma controlada y segura (Anthropic, 2024). Paralelamente, Data API Builder proporciona un mecanismo para exponer entidades de bases de datos mediante interfaces gobernadas, permitiendo definir de forma explícita los recursos, operaciones y permisos disponibles para cada consumidor (Microsoft, 2025). La integración de ambas tecnologías posibilita que los agentes consulten información empresarial a través de herramientas predefinidas, evitando la construcción directa de consultas por parte del modelo de lenguaje. Diversos estudios han señalado que la incorporación de mecanismos de control de acceso y capas intermedias de abstracción contribuye significativamente a reducir la superficie de ataque y mejorar la gobernanza de los datos en entornos donde intervienen sistemas de inteligencia artificial (NIST, 2024; Microsoft, 2025). En este contexto, las arquitecturas basadas en herramientas expuestas mediante protocolos estandarizados representan una alternativa prometedora para habilitar el acceso seguro a información corporativa sin comprometer la integridad de los sistemas transaccionales.

La presente investigación se enmarca en esta tendencia tecnológica y propone la implementación de un servidor basado en el Protocolo de Contexto de Modelo utilizando Data API Builder para

exponer de manera segura una base de datos transaccional alojada en Azure SQL Database. La solución emplea SQL Server Management Studio para el diseño y administración del esquema relacional y Data API Builder desplegado en Azure para publicar las entidades de negocio como herramientas consumibles por agentes compatibles con el protocolo. El objetivo general de esta investigación es diseñar, implementar y evaluar un servidor basado en el Protocolo de Contexto de Modelo mediante Data API Builder que permita a usuarios no técnicos y agentes de inteligencia artificial acceder a información contenida en bases de datos transaccionales de forma segura, gobernada y eficiente, eliminando la necesidad de generar consultas de lenguaje de consulta estructurado de manera manual o mediante traducción directa desde lenguaje natural.

2. METODOLOGÍA

2.1. LEVANTAMIENTO DE REQUERIMIENTOS

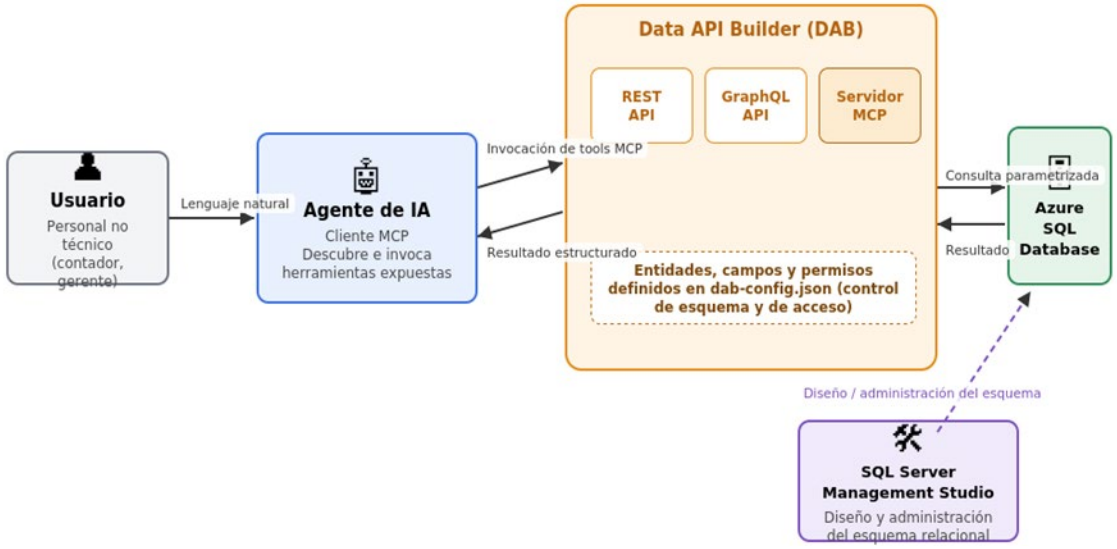
Para identificar las necesidades funcionales y de seguridad del sistema propuesto, se realizaron entrevistas semiestructuradas a profesionales de áreas no técnicas pertenecientes al sector financiero, incluyendo contadores, administradores y gerentes. El propósito de esta fase fue comprender las dificultades existentes en el acceso a información corporativa y determinar los requisitos necesarios para una solución basada en inteligencia artificial. Las entrevistas fueron guiadas mediante un protocolo de preguntas abiertas que permitió explorar tanto las prácticas actuales de consulta de información como las percepciones sobre seguridad y accesibilidad de los datos. Entre las preguntas formuladas se incluyeron las siguientes:

- ¿Qué tipo de información consulta frecuentemente?
- ¿Cuál es su nivel de familiaridad con el lenguaje de consulta estructurado?
- ¿Qué dificultades enfrenta al acceder a datos mediante los sistemas actuales?
- ¿Qué preocupaciones tendría respecto a la seguridad de los datos si un agente de inteligencia artificial accediera directamente a ellos?

Todas las entrevistas fueron realizadas con consentimiento informado de los participantes, registradas para su posterior análisis, transcritas manualmente y sometidas a un proceso de codificación temática que permitió identificar patrones comunes en las respuestas. La duración promedio de cada entrevista fue de treinta minutos. A partir del análisis realizado, se identificaron los principales requerimientos de la solución, destacándose la necesidad de realizar consultas en lenguaje natural, acceder a información empresarial sin conocimientos técnicos especializados, garantizar mecanismos robustos de seguridad y mantener un modelo de control de acceso basado en permisos y roles.

Figura 1

Arquitectura del servidor MCP construido con Data API Builder sobre Azure SQL Database para el acceso seguro de agentes de inteligencia artificial



Fuente: *Elaboración propia.*

2.2. DISEÑO DE LA SOLUCIÓN

La arquitectura de la solución fue diseñada considerando tres componentes fundamentales:

a) **Base de datos relacional.** Se diseñó y administró mediante SQL Server Management Studio, utilizando un modelo representativo de un sistema transaccional bancario. El esquema incluyó entidades relacionadas con clientes, cuentas, transacciones y préstamos, incorporando claves primarias, claves foráneas y restricciones de integridad referencial para garantizar la consistencia de los datos.

b) **Capa de acceso mediante Data API Builder.** Se configuró utilizando un archivo declarativo de configuración que define las entidades expuestas, los campos visibles y los permisos asociados a cada rol de usuario. Esta capa genera automáticamente interfaces de acceso y un servidor compatible con el Protocolo de Contexto de Modelo, permitiendo exponer únicamente aquellos recursos previamente autorizados.

c) **Agente de inteligencia artificial compatible con el Protocolo de Contexto de Modelo.** El agente actúa como cliente del protocolo, identificando dinámicamente las herramientas publicadas por la capa de acceso e invocándolas para responder consultas formuladas en lenguaje natural. De esta manera, las solicitudes son atendidas sin necesidad de generar consultas de lenguaje de consulta estructurado de forma directa.

La arquitectura resultante permite desacoplar la capa de inteligencia artificial de la lógica de acceso a datos, estableciendo una capa intermedia encargada de la validación de operaciones, la aplicación de permisos y la gobernanza de la información.

2.3. DESARROLLO E IMPLEMENTACIÓN

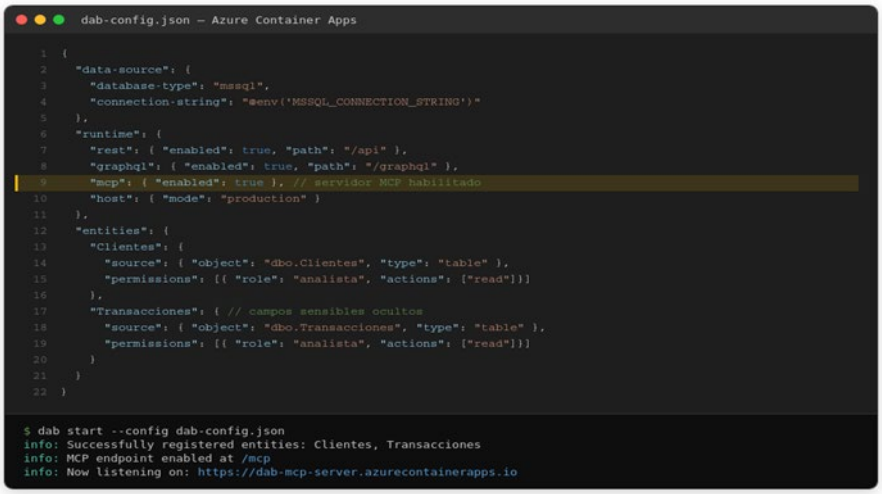
La solución fue implementada sobre la plataforma Microsoft Azure. En la primera etapa se creó una base de datos relacional en Azure SQL Database mediante SQL Server Management Studio, definiendo las tablas, relaciones, índices y usuarios requeridos para representar un entorno transaccional empresarial. Posteriormente, se desplegó Data API Builder y se habilitó la compatibilidad con el Protocolo de Contexto de Modelo. Se configuraron las entidades correspondientes a las tablas de la base de datos, definiendo los campos visibles, las operaciones permitidas y los roles autorizados para cada recurso. Esta configuración permitió delegar en la capa de acceso el control del esquema de datos y la gestión de permisos, evitando que tales responsabilidades fueran asumidas por el modelo de lenguaje.

Con el propósito de garantizar la reproducibilidad del estudio, se documentó detalladamente cada una de las etapas de implementación, incluyendo la estructura de la base de datos, la configuración de entidades y permisos, así como los parámetros utilizados durante el despliegue en la nube. Toda la documentación técnica, el código fuente y los archivos de configuración empleados fueron almacenados en un repositorio público disponible en GitHub, permitiendo la replicación y validación independiente de la propuesta. Repositorio del proyecto:

https://github.com/CarlaMamaniChavez/MCP-database-Article

Figura 2

Configuración del servidor MCP en Data API Builder y habilitación del acceso seguro sobre Azure SQL Database



Fuente: *Elaboración propia.*

2.4. VALIDACIÓN DEL SISTEMA

Para evaluar el desempeño funcional y la efectividad de los mecanismos de seguridad implementados, se diseñó un conjunto de treinta casos de prueba compuestos por solicitudes en lenguaje natural elaboradas por usuarios sin conocimientos técnicos en lenguaje de consulta estructurado. Las pruebas fueron ejecutadas utilizando un agente de inteligencia artificial conectado como cliente MCP al servidor publicado mediante Data API Builder. Los escenarios evaluados incluyeron consultas con filtros simples, condiciones múltiples, ordenamiento de resultados, operaciones de agregación y recuperación de información desde distintas entidades del sistema. Adicionalmente, se incorporaron pruebas orientadas a verificar la robustez de los controles de seguridad mediante intentos deliberados de acceso a entidades, atributos u operaciones no autorizadas. Estas pruebas permitieron comprobar la correcta aplicación de las políticas de permisos y restricciones definidas en la configuración del sistema.

Los resultados fueron registrados en una hoja de validación diseñada para documentar la herramienta invocada, la respuesta obtenida y la correspondencia entre el resultado esperado y el resultado entregado por el sistema. Asimismo, se verificó que todos los intentos de acceso no autorizado fueran rechazados adecuadamente por los mecanismos de control implementados. Los

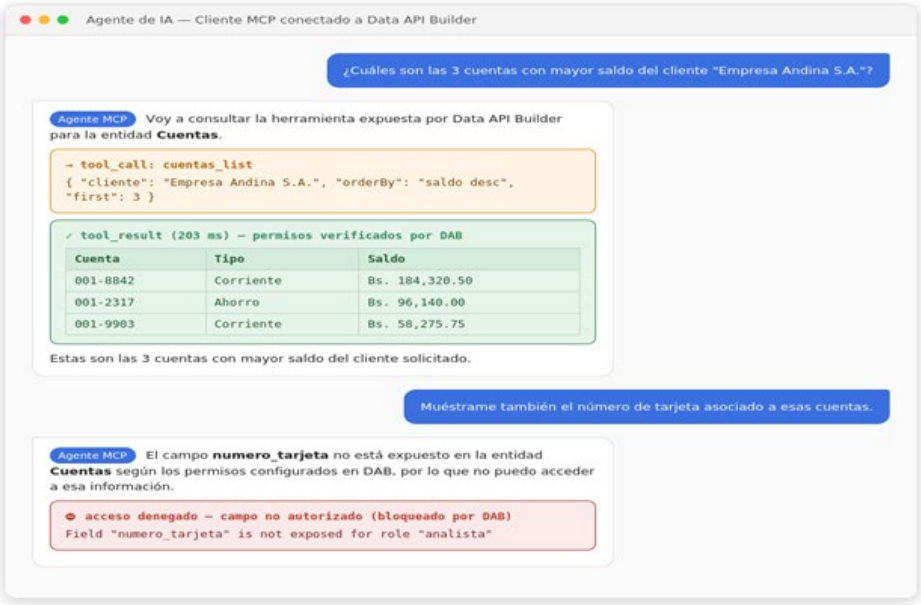
datos obtenidos permitieron medir la precisión de las respuestas, la eficiencia de la solución y el nivel de cumplimiento de los requisitos de seguridad establecidos durante el diseño del sistema.

3. RESULTADOS

El sistema fue sometido a un conjunto de pruebas funcionales con el fin de verificar su desempeño en la resolución de solicitudes en lenguaje natural mediante la invocación de herramientas MCP expuestas por Data API Builder, dentro del contexto de una base de datos transaccional alojada en Azure SQL Database. Las pruebas fueron ejecutadas por usuarios no técnicos, quienes formularon preguntas a un agente de IA conectado como cliente MCP. Cada solicitud fue procesada por el agente, que identificó y ejecutó la herramienta MCP correspondiente, devolviendo los resultados obtenidos directamente desde la base de datos sin generar instrucciones SQL manualmente. Posteriormente, estas salidas fueron validadas mediante comparación con los resultados esperados, definidos previamente por un experto humano en la estructura de la base de datos.

Figura 3

Ejecución de un agente de IA como cliente MCP invocando las herramientas expuestas por Data API Builder en respuesta a una solicitud en lenguaje natural



Fuente: *Elaboración propia.*

La Figura 3 muestra la ejecución del agente de IA con capacidad de resolución de herramientas MCP, transformando entradas en lenguaje natural en invocaciones válidas sobre Data API Builder. Se evidencia la correcta interacción entre el agente, la capa MCP y el motor de base de datos. Los datos obtenidos se resumen en el Cuadro 1, donde se presentan las métricas más relevantes del proceso de validación: número de solicitudes resueltas correctamente mediante herramientas MCP, porcentaje de intentos de acceso no autorizado bloqueados, y tiempo promedio de respuesta del agente.

Tabla 1

Resultados de evaluación de resolución de solicitudes, seguridad y tiempo de respuesta

MÉTRICA EVALUADA	VALOR OBTENIDO
Total, de solicitudes evaluadas	30
Solicitudes resueltas correctamente mediante herramientas MCP	28
Tasa de resolución correcta	93.3 %
Intentos de acceso no autorizado bloqueados	100 % (5/5)
Tiempo promedio de respuesta	1.2 segundos

Fuente: *Elaboración propia.*

En términos de rendimiento, el sistema alcanzó una tasa de resolución correcta del 93.3%, lo que indica que, en 28 de los 30 casos evaluados, el agente identificó e invocó la herramienta MCP adecuada, obteniendo el resultado esperado desde la base de datos. Adicionalmente, se ejecutaron 5 intentos deliberados de acceso no autorizado (consultas a campos ocultos, entidades sin permiso de lectura y parámetros manipulados para forzar accesos fuera del alcance configurado), los cuales fueron bloqueados en el 100% de los casos por la capa de permisos de Data API Builder, sin que se registrara ninguna fuga de esquema ni ejecución de instrucciones no autorizadas. El tiempo promedio de respuesta se mantuvo en 1.2 segundos, considerando desde el envío de la solicitud hasta la visualización del resultado en pantalla, una mejora respecto a los enfoques basados en generación dinámica de SQL, dado que Data API Builder resuelve las operaciones mediante consultas parametrizadas predefinidas en lugar de sentencias construidas dinámicamente por un modelo de lenguaje. Se identificaron dos casos con errores, asociados a ambigüedades en el lenguaje utilizado por los usuarios que dificultaron la selección de la herramienta MCP correcta por parte del agente. Estas ambigüedades incluían términos imprecisos o solicitudes que combinaban múltiples entidades. Sin embargo, en ninguno de los casos se produjo una caída del servicio, una fuga del esquema de la base de datos ni una ejecución no autorizada; el sistema respondió con un mensaje de error controlado o solicitó una aclaración

adicional. Durante las pruebas, se documentaron distintos tipos de solicitudes resueltas exitosamente, incluyendo:

- Filtrado por fechas y nombres
- Cálculo de totales y promedios
- Ordenamientos ascendentes y descendentes
- Recuperación de registros específicos por condición

El sistema mantuvo un comportamiento estable en todos los escenarios, sin evidencias de sobrecarga ni pérdida de datos. Estos resultados confirman que la arquitectura basada en un servidor MCP construido con Data API Builder, junto con un esquema administrado mediante SQL Server Management Studio, resulta adecuada para el objetivo planteado, ofreciendo un balance favorable entre accesibilidad, gobierno de datos y seguridad.

4. DISCUSIÓN

Esta investigación demuestra que es técnicamente viable implementar un servidor MCP basado en Data API Builder que permita a agentes de inteligencia artificial acceder a bases de datos transaccionales de forma segura y gobernada, sin requerir la generación dinámica de instrucciones SQL por parte de un modelo de lenguaje. A través de un flujo modular basado en tecnologías declarativas como Data API Builder, y un esquema administrado directamente mediante SQL Server Management Studio, los resultados deben entenderse como un paso hacia la democratización segura del acceso a los datos operativos. La alta tasa de resolución correcta alcanzada (93.3%) junto con el bloqueo total de los intentos de acceso no autorizado, sugiere que delegar el control del esquema y los permisos en una capa declarativa como DAB, en lugar de en el propio modelo de lenguaje, reduce significativamente la superficie de ataque sin sacrificar la experiencia conversacional. Este hallazgo se distingue de investigaciones recientes que proponen marcos basados en LLMs para la generación directa de consultas SQL en entornos financieros, como FinSQL (Zhang et al., 2024), en las cuales la validez y seguridad de la consulta dependen en gran medida de la capacidad del modelo de lenguaje para interpretar correctamente el esquema y evitar instrucciones peligrosas. El enfoque aquí propuesto traslada esa responsabilidad al servidor MCP, que expone únicamente operaciones y campos explícitamente autorizados, independientemente de las capacidades o limitaciones del modelo de lenguaje utilizado como cliente.

5. CONCLUSIONES

Este estudio evidenció que es posible implementar un servidor MCP basado en Data API Builder que permita a usuarios no técnicos y a agentes de inteligencia artificial consultar bases de datos transaccionales mediante lenguaje natural, con altos niveles de resolución correcta (93.3%) y un bloqueo total de los intentos de acceso no autorizado, sin necesidad de generar instrucciones SQL manualmente. La elección del enfoque cuantitativo-experimental de tipo aplicado fue determinante para validar rigurosamente la efectividad y seguridad del sistema propuesto. Esta metodología permitió estructurar pruebas controladas con usuarios reales del perfil objetivo, así como pruebas específicas de seguridad, generando datos medibles y replicables sobre el desempeño y la robustez de la herramienta. Gracias a este enfoque, no solo se logró el desarrollo de una solución funcional, sino también la evaluación sistemática de su impacto en la autonomía de consulta, el tiempo de acceso a la información, la reducción de dependencia técnica y, particularmente, el fortalecimiento de la seguridad frente a enfoques tradicionales de generación de SQL mediante modelos de lenguaje. La aplicación de pruebas cuantificables permitió comprobar que el sistema responde de forma estable, precisa y segura en escenarios reales de consulta empresarial. Además, se identificaron limitaciones propias de la interpretación del lenguaje natural por parte del agente MCP, como ambigüedad en las instrucciones de entrada al combinar múltiples entidades, lo que abre líneas de mejora futura mediante la definición de herramientas MCP más granulares y descripciones más precisas de cada entidad.

En conclusión, el enfoque metodológico adoptado no solo sustentó el diseño y validación técnica de la solución, sino que aportó evidencia empírica sobre su viabilidad, utilidad, seguridad y potencial de escalabilidad en entornos empresariales reales. La investigación demuestra así que los servidores MCP basados en Data API Builder pueden ser una vía efectiva para democratizar el acceso a información estructurada de forma segura, especialmente en contextos donde el conocimiento técnico no está generalizado y donde la exposición no controlada de las bases de datos representa un riesgo inaceptable.

6. REFERENCIAS BIBLIOGRÁFICAS

Zhang, C., Mao, Y., Fan, Y., Mi, Y., Gao, Y., Chen, L., Lou, D., & Lin, J. (2024). FinSQL: Model-Agnostic LLMs-based Text-to-SQL Framework for Financial Analysis. arXiv preprint arXiv:2401.10506. https://doi.org/10.48550/arXiv.2401.10506

Microsoft. (2024). Data API Builder documentation. Microsoft Learn. https://learn.microsoft.com/en-us/azure/data-api-builder/

Microsoft. (2024). Azure SQL documentation. Microsoft Learn. https://learn.microsoft.com/en-us/azure/azure-sql/?view=azuresql

Microsoft. (2024). SQL Server Management Studio (SSMS) documentation. Microsoft Learn. https://learn.microsoft.com/en-us/sql/ssms/

Anthropic. (2024). Model Context Protocol specification. https://modelcontextprotocol.io