

Modelos SQL no Azure

Característica	SQL em VM no Azure	Instância Gerenciada de SQL do Azure	Banco de Dados SQL do Azure
Modelo de Serviço	IaaS	PaaS	PaaS
Gerenciamento	SO, SQL Server, Patches (Usuário)	SO, SQL Server, Patches (Azure)	SO, SQL Server, Patches (Azure)
Controle	Total sobre SO e SQL Server	Controle no nível da instância	Controle no nível do banco de dados
Compatibilidade SQL	Mais alta (nível de SO e instância)	Muito alta (nível de instância)	Alta (foco no banco de dados)
Recursos de Instância	Sim (SQL Agent, DB Mail, etc.)	Sim (SQL Agent, DB Mail, etc.)	Limitado/Não disponível
Acesso ao SO	Sim	Não	Não
Rede Virtual (VNet)	Sim (configurável pelo usuário)	Sim (implantação nativa em VNet)	Sim (com Service Endpoints ou Private Link)
Escalabilidade	Manual (tamanho da VM)	Dinâmica (vCores, armazenamento)	Dinâmica (DTUs, vCores, armazenamento)
Alta Disponibilidade	Configurável pelo usuário (ex: Always On)	Integrada	Integrada
Backups	Gerenciados pelo usuário (opções Azure)	Automatizados	Automatizados
Ideal para	"Lift-and-shift", controle total	Modernização, alta compatibilidade PaaS	Novas apps na nuvem, escalabilidade

Tipos de Arquivos

Característica	CSV (Comma Separated Values)	JSON (JavaScript Object Notation)	Parquet	ORC (Optimized Row Columnar)	Avro
Estrutura	Texto plano, delimitado por vírgulas (ou outros)	Texto plano, pares chave-valor, hierárquico	Binário, colunar compactado	Binário, colunar	Binário, baseado em linha, com esquema JSON
Esquema	Sem esquema embutido (requer interpretação externa)	Auto-descritivo, flexível	Esquema embutido nos metadados do arquivo	Esquema embutido nos metadados do arquivo	Esquema definido em JSON, armazenado com os dados ou externamente
Evolução do Esquema	Difícil, pode quebrar leitores	Flexível, mas pode exigir tratamento no código	Suportado (adição de colunas, etc.)	Suportado (adição de colunas, etc.)	Robusto, lida bem com esquemas em evolução
Performance de Leitura	Lenta para grandes volumes, lê linhas inteiras	Relativamente lenta para grandes volumes, parsing	Rápida para consultas analíticas (lê só colunas necessárias)	Rápida para consultas analíticas (lê só colunas necessárias)	Rápida para leitura de linhas inteiras
Performance de Escrita	Rápida para dados simples	Relativamente rápida	Moderada (mais complexa devido à estrutura colunar)	Moderada (mais complexa devido à estrutura colunar)	Rápida
Tipos de Dados Suportados	Principalmente strings, números simples	Strings, números, booleanos, arrays, objetos aninhados	Tipos de dados ricos e aninhados	Tipos de dados ricos e aninhados	Tipos de dados ricos e primitivos
Integração com Big Data	Limitada para grandes datasets	Comum para APIs, mas menos eficiente para analytics	Excelente (Spark, Hadoop, Synapse Analytics)	Excelente (Hive, Spark, Hadoop, Synapse Analytics)	Muito boa (Kafka, Spark, Hadoop)
Casos de Uso Comuns	Pequenos datasets, exportação/importação simples, planilhas	APIs web, arquivos de configuração, dados semiestruturados	Data lakes, data warehousing, processamento analítico em larga escala	Data lakes, data warehousing (especialmente com Hive), processamento analítico	Serialização de dados, streaming de dados (Kafka), armazenamento de dados em linha

ACID

A - Atomicidade (Atomicity)

- O que é: Garante que todas as operações dentro de uma transação sejam concluídas com sucesso como uma unidade única e indivisível. Ou tudo acontece, ou nada acontece.
 - Exemplo: Em uma transferência bancária, a operação de debitar da conta A e creditar na conta B deve ser atômica. Se o débito ocorrer, mas o crédito falhar (por exemplo, por uma queda de energia), a transação inteira é desfeita (rollback), e o dinheiro retorna para a conta A. Nenhum estado intermediário é permitido.
 - Importância para DP-900: Entender como a atomicidade previne dados parciais ou inconsistentes após falhas.
-

C - Consistência (Consistency)

- O que é: Garante que uma transação leve o banco de dados de um estado válido para outro estado válido. Todas as regras definidas para o banco de dados (como restrições, chaves estrangeiras, gatilhos) devem ser respeitadas.
 - Exemplo: Se uma regra de negócios diz que o saldo de uma conta não pode ser negativo, uma transação que tente sacar mais dinheiro do que o disponível será impedida, mantendo a consistência dos dados.
 - Importância para DP-900: Reconhecer que a consistência assegura que os dados sempre sigam as regras de negócios e integridade definidas.
-

I - Isolamento (Isolation)

- O que é: Garante que transações concorrentes (múltiplas transações ocorrendo ao mesmo tempo) não interfiram umas nas outras. Cada transação deve parecer que está sendo executada sozinha no sistema.
 - Exemplo: Se duas pessoas tentam reservar o último assento em um voo simultaneamente, o isolamento garante que apenas uma transação seja bem-sucedida. A outra transação verá os dados como eram antes da primeira transação começar ou somente após ela ser concluída, evitando uma reserva dupla.
 - Importância para DP-900: Compreender como o isolamento previne problemas como leituras sujas (dirty reads), leituras não repetíveis (non-repeatable reads) e leituras fantasmas (phantom reads) em ambientes multiusuário.
-

D - Durabilidade (Durability)

- O que é: Garante que, uma vez que uma transação tenha sido confirmada (commit), suas alterações persistirão e não serão perdidas, mesmo em caso de falhas do sistema (como queda de energia ou crash do servidor).
- Exemplo: Após você receber a confirmação de que sua compra online foi concluída, a durabilidade assegura que essa transação esteja permanentemente registrada no banco de dados e não será desfeita, mesmo que o servidor do site falhe momentos depois.
- Importância para DP-900: Saber que a durabilidade garante a permanência dos dados confirmados, geralmente através de escrita em armazenamento não volátil e logs de transação.

Funções de dados

Papel	Foco Principal	O que eles FAZEM com os dados principalmente?	Exemplo de Ferramenta/Serviço Chave no Azure (Contexto DP-900)
Analista de Dados	Encontrar insights e contar histórias com dados	Analisa, visualiza, cria relatórios	Power BI, Banco de Dados SQL do Azure (para consulta)
Engenheiro de Dados	Construir e manter o fluxo e armazenamento de dados	Ingesta, transforma, armazena, prepara para o consumo	Azure Data Factory, Azure Data Lake Storage, Azure Synapse Analytics
Administrador de BD (DBA)	Garantir que os bancos de dados funcionem bem	Gerencia, protege, otimiza, faz backup	Banco de Dados SQL do Azure, Instância Gerenciada de SQL
Cientista de Dados	Fazer previsões e descobertas usando dados	Constrói modelos de Machine Learning, aplica estatística	Azure Machine Learning (conceitualmente), dados do Azure Data Lake

Característica	Analista de Dados	Engenheiro de Dados	Administrador de BD (DBA)	Cientista de Dados
Objetivo Principal	Interpretar dados, gerar insights	Construir e manter pipelines de dados	Gerenciar e proteger bancos de dados	Prever e otimizar usando modelos
Foco Temporal	Passado e Presente ("O que aconteceu?")	Infraestrutura para o fluxo de dados	Saúde e disponibilidade do BD	Futuro e Descoberta ("O que pode acontecer?", "Como podemos fazer acontecer?")
Tipo de Análise	Descritiva, Diagnóstica	N/A (Foco em infraestrutura)	N/A (Foco em operações)	Preditiva, Prescritiva, Exploratória
Habilidade Chave	Visualização, SQL, Comunicação	Programação, ETL, Arquitetura de Dados	Gerenciamento de SGBD, Segurança	Machine Learning, Estatística, Modelagem
Ferramenta Principal (Exemplo Azure)	Power BI, Azure Synapse Analytics (consultas)	Azure Data Factory, Azure Databricks, Azure Storage	Azure SQL Database, SQL Server em VM	Azure Machine Learning, Azure Databricks

OLAP x OLTP

Característica	OLTP (Processamento de Transações Online)	OLAP (Processamento Analítico Online)
Objetivo	Executar operações diárias do negócio	Suportar a tomada de decisões
Operações	Leituras, inserções, atualizações, exclusões frequentes	Principalmente leituras, consultas complexas
Estrutura de Dados	Normalizada (3FN+)	Desnormalizada (esquema estrela/floco de neve)
Fonte de Dados	Dados operacionais atuais	Dados históricos agregados
Volume de Dados	Transações individuais, dados detalhados	Grandes volumes de dados agregados
Usuários	Operacionais, aplicações	Analistas, gerentes, executivos
Foco	Eficiência transacional, concorrência	Análise de dados, insights
Unidade de Trabalho	Transações curtas e rápidas	Consultas complexas e demoradas
Prioridade	Disponibilidade, velocidade da transação	Velocidade da consulta, flexibilidade
Exemplo Típico	Sistema de caixa de supermercado	Sistema de análise de vendas trimestral

Databricks

Característica	Descrição
Plataforma	Unificada de Análise de Dados (Unified Analytics Platform) - PaaS (Plataforma como Serviço) no Azure, conhecida como Azure Databricks.
Base Tecnológica	Construída sobre Apache Spark, um poderoso motor de processamento distribuído de código aberto para big data e machine learning.
Componentes Chave	Integra Delta Lake (camada de armazenamento que traz confiabilidade ACID para data lakes), MLflow (gerenciamento do ciclo de vida de machine learning), e notebooks colaborativos.
Linguagens Suportadas	Principalmente Python, Scala, SQL e R, permitindo que diferentes equipes colaborem usando suas linguagens preferidas.
Casos de Uso	Engenharia de dados (ETL/ELT), análise exploratória de dados, machine learning em larga escala, processamento de streaming de dados, e business intelligence.
Colaboração	Oferece notebooks interativos que permitem que múltiplos usuários trabalhem juntos em tempo real, compartilhando código, visualizações e texto.

Integração no Azure	Profundamente integrado com serviços do Azure como Azure Data Lake Storage, Azure Blob Storage, Azure Synapse Analytics, Power BI, e Azure Machine Learning.
Gerenciamento	Simplifica o gerenciamento de clusters Spark, automatizando a configuração, o dimensionamento e a manutenção.

Tipos de Bancos de Dados

Característica	Bancos de Dados Relacionais (SQL)	Bancos de Dados de Chave-Valor (NoSQL)	Bancos de Dados de Grafos (NoSQL)	Bancos de Dados de Documento (NoSQL)	Bancos de Dados Colunares (NoSQL)
Modelo de Dados	Tabelas com linhas e colunas; relacionamentos definidos por chaves estrangeiras. 	Dicionário simples de pares chave-valor. A "chave" é um identificador único para um "valor".  	Nós (entidades) e Arestas (relacionamentos diretos entre nós). As arestas têm propriedades. 	Documentos flexíveis, frequentemente em formatos como JSON, BSON ou XML. Cada documento é auto-contido. 	Dados organizados em famílias de colunas (colunas agrupadas) em vez de linhas. 
Esquema	Rígido (schema-on-write): a estrutura da tabela é definida antes da inserção dos dados. 	Sem esquema ou esquema flexível (schema-on-read): a estrutura do valor pode variar. 	Sem esquema ou esquema flexível para as propriedades dos nós e arestas. 	Flexível (schema-on-read): a estrutura pode variar entre documentos dentro da mesma coleção. 	Flexível: novas colunas podem ser adicionadas a famílias de colunas dinamicamente. 
Escalabilidade	Principalmente vertical (aumentando os recursos de um único servidor). A escalabilidade horizontal é mais complexa. 	Tipicamente horizontal (distribuindo dados entre múltiplos servidores). 	Tipicamente horizontal, mas a natureza conectada dos dados pode apresentar desafios em distribuições massivas. 	Tipicamente horizontal, permitindo a distribuição de documentos entre servidores. 	Tipicamente horizontal, projetado para distribuir grandes volumes de dados em clusters. 

<p>Consistência</p>	<p>Forte consistência (ACID: Atomicidade, Consistência, Isolamento, Durabilidade). ✅</p>	<p>Geralmente consistência eventual (Eventually consistent). Alguns oferecem consistência configurável. 🔄</p>	<p>Varia, mas muitos podem oferecer transações ACID para operações de grafos. Alguns seguem BASE. 🔄</p>	<p>Muitos oferecem ACID no nível do documento. A consistência entre documentos pode ser eventual (BASE). 📄 / 🔄</p>	<p>Geralmente consistência eventual (BASE) ou consistência ajustável por operação. 🔄</p>
<p>Linguagem de Consulta</p>	<p>SQL (Structured Query Language). 📄</p>	<p>APIs simples (GET, PUT, DELETE) baseadas na chave. Consultas complexas sobre valores são limitadas. 💻</p>	<p>Linguagens específicas para grafos (ex: Cypher, Gremlin, SPARQL) focadas em percorrer relacionamentos. 📖</p>	<p>APIs específicas do banco de dados ou linguagens de consulta semelhantes ao JSON (ex: MongoDB Query Language). 🗣️</p>	<p>Linguagens de consulta semelhantes ao SQL (ex: CQL para Cassandra) ou APIs específicas. 🗣️</p>
<p>Integridade dos Dados</p>	<p>Alta, garantida por restrições, chaves primárias e estrangeiras. 🛡️</p>	<p>Menor ênfase na integridade relacional, mais na disponibilidade e velocidade. 🚀</p>	<p>Mantém a integridade dos relacionamentos diretos. Pode ter restrições. 🔗</p>	<p>Integridade garantida no nível do documento. Relacionamentos entre documentos são gerenciados pela aplicação. 📄</p>	<p>Menor ênfase na integridade relacional. A consistência é gerenciada pela aplicação ou configurável. 🚫</p>
<p>Performance</p>	<p>Bom para consultas complexas com JOINS e transações. Pode ser um gargalo com volumes muito grandes ou escritas intensas. 🏃</p>	<p>Extremamente rápido para leituras e escritas simples por chave (baixa latência). ⚡</p>	<p>Otimizado para consultas que exploram relacionamentos e caminhos (consultas de "vários saltos"). 👟</p>	<p>Bom para ler e escrever documentos inteiros. Flexível para dados que não se encaixam bem em tabelas. ⚙️</p>	<p>Otimizado para leituras e escritas rápidas de grandes volumes de dados, e para consultas que agregam valores de colunas específicas. ⚡ 📊</p>

<p>Casos de Uso Típicos</p>	<p>Sistemas transacionais (OLTP), aplicações financeiras, sistemas que exigem alta consistência e dados estruturados.</p> <p> ERP</p>	<p>Caching, gerenciamento de sessões de usuário, dados que podem ser acessados rapidamente por um identificador único.</p> <p> </p>	<p>Redes sociais, sistemas de recomendação, detecção de fraudes, gerenciamento de redes, dados com muitos relacionamentos complexos.</p> <p>  </p>	<p>Catálogos de produtos, gerenciamento de conteúdo, perfis de usuário, aplicações móveis, dados semiestruturados</p> <p> </p>	<p>Data warehouses, Business Intelligence (BI), análise de séries temporais, Internet das Coisas (IoT), alta taxa de escrita.</p> <p> </p>
<p>Exemplos</p>	<p>MySQL, PostgreSQL, SQL Server, Oracle Database, Azure SQL Database.</p> <p> </p>	<p>Redis, Amazon DynamoDB (modo chave-valor), Memcached, Azure Cache for Redis.</p> <p> </p>	<p>Neo4j, Amazon Neptune, Azure Cosmos DB (API Gremlin).</p> <p> </p>	<p>MongoDB, Couchbase, Azure Cosmos DB (API NoSQL / Core).</p> <p> </p>	<p>Apache Cassandra, Google Bigtable, Azure Cosmos DB (API Cassandra), Apache HBase.</p> <p> </p>

Data lake x Data Warehouse

Característica	Data Lake	Data Warehouse
Dados Armazenados	Todos os tipos de dados: brutos, não processados, semiestruturados (JSON, XML, logs), estruturados e não estruturados (imagens, vídeos, áudio). 🛒	Principalmente dados estruturados e processados, que foram limpos, transformados e modelados para um propósito específico. 📊
Esquema	Schema-on-Read: O esquema é aplicado quando os dados são lidos e analisados, não quando são armazenados. Alta flexibilidade. 🏹	Schema-on-Write: O esquema é definido antes que os dados sejam carregados. Os dados devem estar em conformidade com esse esquema. 📐
Processamento	Os dados são carregados em seu formato original (ELT - Extract, Load, Transform, se necessário, no momento da análise). 🔄	Os dados passam por um processo de ETL (Extract, Transform, Load) antes de serem armazenados, garantindo qualidade e formato. ✅
Finalidade Principal	Exploração de dados, descoberta de insights, data science, machine learning, análise de dados brutos. 🔬🧪	Business Intelligence (BI), relatórios corporativos, dashboards, análise histórica e tomada de decisão baseada em dados já refinados. 📈📄
Usuários Típicos	Cientistas de dados, engenheiros de dados, analistas de dados (que trabalham com dados brutos). 👨‍🔬👨‍💻	Analistas de negócios, usuários de negócios, gestores. 👨‍💼👩‍💼
Agilidade e Flexibilidade	Alta. Fácil de ingerir novos tipos de dados sem necessidade de definir estruturas previamente. Adaptável a mudanças rápidas. 🚀	Menor. Alterações no esquema podem ser complexas e demoradas, pois exigem a redefinição de estruturas e processos de ETL. 🤖
Custo de Armazenamento	Geralmente mais baixo, pois utiliza armazenamento de objetos de baixo custo (ex: Azure Data Lake Storage, Amazon S3) para grandes volumes de dados brutos. 💰	Geralmente mais alto, devido ao armazenamento de dados processados e otimizados em sistemas de banco de dados mais caros e à necessidade de manter dados históricos estruturados. 💳
Velocidade de Análise	Pode variar. Consultas em dados brutos podem ser mais lentas se não otimizadas. A velocidade depende da	Geralmente mais rápida para relatórios e consultas predefinidas, pois os dados

	ferramenta de análise e da preparação dos dados no momento da leitura. 🌱	já estão otimizados e indexados para esse fim. 🚀
Qualidade dos Dados	Variável. Contém dados brutos que podem ser de qualidade desconhecida até serem processados. Pode haver "pântanos de dados" se não bem governados. 🌊	Alta. Os dados são limpos, validados e transformados antes do carregamento, garantindo uma "fonte única da verdade" confiável. ✨
Exemplos de Tecnologias	Azure Data Lake Storage, AWS S3, Hadoop Distributed File System (HDFS), Google Cloud Storage. Ferramentas de processamento como Apache Spark, Azure Databricks.	Azure Synapse Analytics (componente de Data Warehouse), Google BigQuery, Amazon Redshift, Snowflake, SQL Server (com design de DW).

ETL x ELT

Característica	ETL (Extract, Transform, Load)	ELT (Extract, Load, Transform)
Ordem das Etapas	Extrair -> Transformar -> Carregar	Extrair -> Carregar -> Transformar
Onde a Transformação Ocorre	Em um servidor/área de staging intermediário	No sistema de destino (Data Lake, Data Warehouse moderno)
Dados Carregados no Destino	Dados processados e estruturados	Dados brutos ou minimamente processados
Esquema na Ingestão	Schema-on-Write (esquema definido antes de carregar)	Schema-on-Read (esquema aplicado ao ler/transformar)
Flexibilidade	Menor (transformações definidas antecipadamente)	Maior (transformações conforme a necessidade)
Velocidade de Ingestão	Mais lenta (devido à transformação prévia)	Mais rápida (carrega dados brutos diretamente)
Adequado Para	Data Warehouses tradicionais, dados estruturados	Data Lakes, Big Data, dados diversos, nuvem
Uso de Recursos	Pode exigir um servidor ETL dedicado	Utiliza os recursos do sistema de destino

Data Lake Storage Gen 2

Para usar o ADLS Gen2, você cria uma conta de Armazenamento do Azure normal e habilita a opção "Namespace hierárquico" durante a criação. Uma vez habilitado, esse recurso não pode ser desabilitado para a conta de armazenamento.

Característica	Azure Data Lake Storage Gen2 (com HNS)	Armazenamento de Blobs do Azure (sem HNS / Namespace Plano)
Namespace	Hierárquico (pastas/arquivos)	Plano (contêineres/blobs)
Operações em Diretórios	Atômicas e rápidas	Lentas (requerem enumeração e operação em cada blob)
ACLs POSIX	Sim, para controle granular	Não diretamente (usa RBAC, SAS, chaves de acesso)
Otimização para Analytics	Alta (driver ABFS)	Menor (embora ainda usável)
Uso Principal	Data lakes, big data analytics	Armazenamento de objetos de propósito geral, backups, mídia

Tipos de Gráficos

Tipo de Gráfico	Uso Principal / Representa
 Gráfico de Barras / Colunas	Comparar valores entre diferentes categorias ou itens discretos.
 Gráfico de Linhas	Mostrar tendências e mudanças ao longo do tempo ou de uma sequência contínua.
 Gráfico de Pizza / Rosca	Mostrar a proporção de cada categoria em relação a um todo (100%). Mais eficaz com poucas categorias.
Gráfico de Dispersão (Scatter Plot)	Identificar a relação ou correlação entre duas variáveis numéricas.
 Histograma	Representar a distribuição de frequência de um conjunto de dados numéricos contínuos dentro de intervalos específicos (bins).
 Mapa de Árvore (Treemap)	Exibir dados hierárquicos como um conjunto de retângulos aninhados, onde a área de cada retângulo é proporcional ao seu valor.
 Matriz (Matrix)	Apresentar dados detalhados de forma organizada em linhas e colunas, frequentemente com agregações, agrupamentos e hierarquias.

Lote x Fluxo

Característica	Processamento em Lote (Batch)	Processamento de Fluxo (Stream)
Dados	Coletados e processados em grandes lotes	Processados continuamente à medida que chegam
Tamanho dos Dados	Grande volume por lote	Pequenos "pedaços" ou eventos individuais
Latência	Alta (horas, dias)	Baixa (milissegundos, segundos)
Tempo de Análise	Agendado, periódico	Em tempo real ou quase real
Estado dos Dados	Geralmente estático durante o processo	Dinâmico, fluxo contínuo
Casos de Uso	Relatórios, folha de pagamento, ETL robusto	Detecção de fraude, IoT, monitoramento real
Ideal para	Análise de dados históricos e grandes volumes de uma vez	Respostas rápidas e insights imediatos

Cosmos DB

Distribuição Global: Você pode distribuir seus dados para qualquer região do Azure com o clique de um botão. Isso aproxima os dados dos seus usuários, reduzindo a latência e aumentando a disponibilidade global.

Escalabilidade Elástica: Permite escalar a taxa de transferência (requisições por segundo) e o armazenamento de forma independente e elástica, pagando apenas pelo que você usa.

Baixa Latência Garantida: Oferece SLAs para latência de leitura e escrita na casa dos milissegundos.

Alta Disponibilidade: Garante 99,999% de disponibilidade para leitura e escrita em contas multirregionais com várias regiões de gravação.

Totalmente Gerenciado (PaaS): O Azure cuida da administração do banco de dados, incluindo patches, atualizações, gerenciamento de capacidade e backups

API	Modelo de Dados Suportado	Descrição / Caso de Uso Comum
API para NoSQL (Core / SQL API)	Documento (JSON)	API nativa do Cosmos DB, ideal para novas aplicações que precisam de um banco de dados de documentos flexível e consultável com SQL.
API do Azure Cosmos DB para MongoDB	Documento (BSON)	Compatível com o protocolo de transmissão do MongoDB. Permite migrar aplicações MongoDB existentes ou construir novas com ferramentas MongoDB.
API do Azure Cosmos DB para PostgreSQL	Relacional	Oferece um serviço PostgreSQL gerenciado e distribuído (baseado no Citus), ideal para aplicações que exigem um modelo relacional com escalabilidade. (Pode estar em versão prévia/preview)
API do Azure Cosmos DB para Apache Cassandra	Colunar	Compatível com o protocolo de transmissão do Cassandra. Permite migrar aplicações Cassandra ou construir novas com ferramentas Cassandra.
API do Azure Cosmos DB para Apache Gremlin	Grafo	Suporta a linguagem de consulta de grafos Gremlin. Usado para modelar e consultar dados com relacionamentos complexos (redes sociais, recomendações).
API do Azure Cosmos DB para Tabela	Chave-Valor	Oferece a mesma API do Armazenamento de Tabelas do Azure, mas com os benefícios do Cosmos DB (distribuição global, latência, taxa de transferência).

Normalização

Minimiza a Redundância de Dados: Evitar que a mesma informação seja armazenada em múltiplos lugares.

Evita Anomalias de Modificação: Prevenir problemas que podem ocorrer ao inserir, atualizar ou excluir dados.

Índices

Um índice em um banco de dados é uma estrutura de dados especial (semelhante ao índice remissivo de um livro) que melhora a velocidade das operações de recuperação de dados em uma tabela. Ele faz isso criando uma cópia ordenada de uma ou mais colunas da tabela, com ponteiros para as linhas correspondentes na tabela original.

Índice Clusterizado (Clustered Index):

- Define a ordem física de armazenamento dos dados na tabela.
- Só pode haver um índice clusterizado por tabela (porque os dados só podem ser armazenados em uma ordem física).
- Geralmente criado na chave primária da tabela.

Índice Não Clusterizado (Non-Clustered Index):

- Cria uma estrutura separada do armazenamento físico dos dados da tabela.
- Contém os valores das colunas indexadas e um ponteiro para a linha correspondente na tabela de dados (que pode ser o índice clusterizado ou um identificador de linha).
- Uma tabela pode ter múltiplos índices não clusterizados.

View (visão/exibição)

Uma view em um banco de dados é essencialmente uma tabela virtual cujo conteúdo é definido por uma consulta SQL armazenada. Diferentemente de uma tabela comum, uma view geralmente não armazena dados fisicamente por si só. Em vez disso, ela representa o resultado de uma consulta que é executada dinamicamente sempre que a view é acessada.

Pense nela como uma "janela" ou um "atalho" para visualizar dados que residem em uma ou mais tabelas subjacentes, de uma maneira específica e predefinida.

SQL

Nome do Comando	Parte do SQL	Breve Resumo da Função
CREATE	DDL (Data Definition Language)	 Cria novos objetos no banco de dados (ex: tabelas, views, índices).
ALTER	DDL (Data Definition Language)	 Modifica a estrutura de objetos existentes no banco de dados.
DROP	DDL (Data Definition Language)	 Remove objetos existentes do banco de dados.
TRUNCATE	DDL (Data Definition Language)	 Remove todas as linhas de uma tabela rapidamente (sem registrar deleções individuais).
SELECT	DQL (Data Query Language)	 Recupera dados de uma ou mais tabelas.
INSERT	DML (Data Manipulation Language)	 Adiciona novas linhas (registros) a uma tabela.
UPDATE	DML (Data Manipulation Language)	 Modifica dados existentes em linhas de uma tabela.
DELETE	DML (Data Manipulation Language)	 Remove linhas de uma tabela.
GRANT	DCL (Data Control Language)	 Concede permissões a usuários ou papéis para acessar objetos do banco de dados.
REVOKE	DCL (Data Control Language)	 Remove permissões previamente concedidas a usuários ou papéis.
COMMIT	TCL (Transaction Control Language)	 Salva permanentemente todas as modificações de dados feitas na transação atual.
ROLLBACK	TCL (Transaction Control Language)	 Desfaz todas as modificações de dados feitas na transação atual ou desde um SAVEPOINT.
SAVEPOINT	TCL (Transaction Control Language)	 Define um ponto dentro de uma transação para o qual se pode reverter parcialmente.

Camadas de Acesso

Característica	Hot (Quente)	Cool (Fria)	Cold (Muito Fria)	Archive (Arquivamento)
Uso Principal / Ideal Para	Dados acessados ou modificados com frequência.	Dados acessados com pouca frequência, armazenados por pelo menos 30 dias.	Dados raramente acessados, armazenados por pelo menos 90 dias.	Dados raramente acessados, armazenados por pelo menos 180 dias, com latência de recuperação de horas.
Custo de Armazenamento	Mais alto	Mais baixo que Hot.	Mais baixo que Cool.	O mais baixo de todas as camadas.
Custo de Acesso / Recuperação	Mais baixo	Mais alto que Hot (taxa de recuperação por GB).	Mais alto que Cool (taxa de recuperação por GB).	O mais alto (custos para reidratar e recuperar).
Latência de Recuperação	Milissegundos (acesso online)	Milissegundos (acesso online)	Milissegundos (acesso online)	Horas (dados offline, precisam ser reidratados)
Duração Mínima de Armazenamento	Não aplicável	30 dias (taxa de exclusão antecipada aplicável)	90 dias (taxa de exclusão antecipada aplicável)	180 dias (taxa de exclusão antecipada aplicável)
Casos de Uso Típicos	Conteúdo frequentemente acessado. Dados em processamento	Backups de curto prazo. Recuperação de desastres	Backups de longo prazo. Arquivos de projetos mais antigos	Arquivamento de longo prazo (conformidade). Dados brutos retidos

Tipos de Blobs

Característica	Blob de Bloco (Block Blob)	Blob de Acréscimo (Append Blob)	Blob de Página (Page Blob)
Descrição Principal	Armazena texto e dados binários. Composto por blocos de dados que podem ser gerenciados individualmente.	Similar aos blobs de bloco, mas otimizado para operações de acréscimo (adição de dados no final).	Coleção de páginas de 512 bytes otimizada para operações de leitura e escrita aleatórias.
Estrutura	Composto por blocos (até 100 MB cada, com um máximo de 50.000 blocos por blob). Cada bloco tem um ID de bloco.	Composto por blocos, mas otimizado para adicionar blocos sequencialmente ao final do blob.	Sequência de páginas de 512 bytes. Permite leituras e escritas em qualquer página.
Operações Principais	Upload de blocos individuais, consolidação de blocos (Put Block List), leitura e escrita de intervalos de bytes.	Acréscimo de blocos (Append Block) ao final do blob. Não permite modificação de blocos existentes.	Leitura e escrita de páginas específicas (acesso aleatório).
Tamanho Máximo	Aproximadamente 4.75 TB (50.000 blocos * 100 MB/bloco). A Microsoft frequentemente atualiza esses limites.	Aproximadamente 195 GB (50.000 blocos * 4 MB/bloco). A Microsoft frequentemente atualiza esses limites.	Até 8 TB.
Otimizado Para	Upload eficiente de grandes volumes de dados (paralelização), streaming de dados, armazenamento de objetos discretos grandes.	Cenários de registro (logging), adição contínua de dados, como dados de sensores ou logs de aplicação.	Operações de I/O aleatórias e frequentes, como discos para Máquinas Virtuais (VMs) do Azure.
Casos de Uso Típicos	Arquivos de mídia (vídeos, imagens, áudio), documentos, backups, arquivos de instalação, dados para análise.	Arquivos de log, dados de telemetria, dados de auditoria, arquivos que são constantemente atualizados com novas informações no final.	Discos de sistema operacional e de dados para VMs do Azure (IaaS Disks), arquivos de banco de dados que exigem acesso aleatório.
Modificabilidade	Blocos individuais podem ser modificados ou substituídos antes da consolidação (Put Block List). Após a consolidação, o blob é tratado como um todo.	Novos dados só podem ser adicionados ao final do blob. Blocos existentes não podem ser modificados ou excluídos.	Páginas individuais podem ser escritas e lidas aleatoriamente.

Tabela fato x Tabela dimensão

Característica	Tabela Fato (Fact Table)	Tabela Dimensão (Dimension Table)
Conteúdo Principal	 Medidas quantitativas e métricas de um evento de negócio (ex: quantidade vendida, valor da venda, contagem de cliques).	descriptive Atributos descritivos que fornecem contexto aos fatos (ex: nome do produto, categoria, data, nome do cliente, região).
Objetivo	Armazenar os "o quês" e "quantos" de um processo de negócio.	Descrever o "quem", "o quê", "onde", "quando", "porquê" e "como" dos fatos.
Estrutura Típica	Geralmente longa (muitas linhas, representando muitos eventos) e estreita (poucas colunas: chaves estrangeiras e medidas).	Geralmente mais curta (menos linhas, representando entidades únicas) e larga (muitas colunas descritivas).
Tipos de Dados Predominantes	Numéricos, aditivos ou semi-aditivos (valores que podem ser somados ou agregados).	Textuais, categóricos, datas, flags booleanas.
Chaves	Contém múltiplas chaves estrangeiras que se conectam às chaves primárias das tabelas dimensão. Pode ter uma chave primária composta (formada pelas chaves estrangeiras) ou uma chave primária substituta.	Contém uma chave primária única (geralmente um identificador substituto, ex: ProdutoID, ClienteID) que é referenciada pela tabela fato.
Granularidade	Define o nível de detalhe de cada registro. Uma linha na tabela fato representa um evento específico no nível mais baixo de detalhe (ex: uma venda individual de um produto).	Cada linha representa uma instância única de uma dimensão (ex: um produto específico, um cliente específico).
Atualizações	Frequentemente atualizada com novos registros à medida que os eventos de negócio ocorrem. Os dados históricos geralmente não são alterados.	Os atributos podem mudar ao longo do tempo (ex: mudança de endereço de um cliente), o que pode ser tratado por técnicas como Dimensões Lentamente Mutáveis (SCD - Slowly Changing Dimensions).
Relacionamento	Localizada no centro de um esquema estrela (star schema) ou floco de neve (snowflake schema), conectada a múltiplas tabelas dimensão.	Cercam a tabela fato no esquema estrela ou floco de neve.

Exemplo de Colunas	DataID, ProdutoID, ClienteID, LojaID, QuantidadeVendida, ValorTotalVenda.	ProdutoID (PK), NomeProduto, CategoriaProduto, MarcaProduto, CorProduto.
Foco na Análise	Utilizada para agregações, cálculos, medições de desempenho e análise de tendências.	Utilizada para filtrar, agrupar e rotular os dados nas consultas e relatórios.

Serviços de dados no Azure

Característica	Azure Data Factory (ADF)	Azure Databricks	Azure Synapse Analytics	Microsoft Fabric	Azure Data Explorer (ADX)
Principal Função	Orquestração de dados e ETL/ELT em nuvem (PaaS).	Plataforma de análise unificada baseada em Apache Spark (PaaS).	Serviço de análise ilimitado que combina data warehousing e análise de Big Data (PaaS).	Plataforma de análise unificada de ponta a ponta em SaaS, integrando Power BI, Data Factory, Synapse e Data Explorer.	Serviço de análise de dados rápido e altamente escalável para telemetria e logs (PaaS).
Principais Casos de Uso	- Ingestão de dados de diversas fontes - Movimentação de dados - Transformação de dados em escala - Orquestração de pipelines	- Engenharia de dados em larga escala - Machine Learning e ciência de dados colaborativa - Processamento de streaming - Análise interativa de Big Data	- Data Warehousing moderno - Análise de Big Data (SQL e Spark) - Exploração de dados em Data Lakes - Integração com Power BI	- Análise de ponta a ponta (de Data Lake a relatórios de BI) - Engenharia de dados e ciência de dados unificadas - Business Intelligence em tempo real	- Análise de logs e telemetria - Monitoramento de séries temporais - Análise interativa de dados de streaming - IoT analytics
Processamento de Dados	Lote (principalmente), micro-lote (com Mapping Data Flows).	Lote, streaming, interativo.	Lote, streaming (via Spark pools), interativo (SQL e Spark).	Lote, streaming, interativo (abrange os recursos dos componentes integrados).	Quase em tempo real, streaming, interativo.
Linguagens Suportadas	JSON (para definição de pipeline), SQL (em data flows),	Python, Scala, SQL, R, Java.	SQL (pools dedicados e serverless), Python, Scala,	SQL, Python, Scala, R, KQL, DAX, etc. (dependendo do	Kusto Query Language (KQL).

	C# (custom activities).		.NET (Spark pools).	componente Fabric).	
Integração com Spark	Não diretamente, mas pode orquestrar atividades do Databricks ou Synapse Spark.	Plataforma nativa do Apache Spark.	Pools Spark integrados para processamento de Big Data.	Inclui o motor Spark através das experiências de Engenharia de Dados e Ciência de Dados (similar ao Synapse Spark).	Não usa Spark nativamente; possui seu próprio motor de indexação e consulta.
Armazenamento de Dados	Conecta-se a diversas fontes e destinos (Azure Blob, Data Lake, SQL DB, etc.). Não armazena dados permanentemente.	Interage com Data Lakes (ADLS Gen2), Blob Storage. O Delta Lake é uma camada de armazenamento fundamental.	Armazena dados em Data Lakes (ADLS Gen2) e em pools SQL dedicados (data warehouse relacional).	OneLake (um Data Lake unificado e lógico para toda a organização, construído sobre ADLS Gen2).	Armazena dados em seu próprio formato otimizado e colunar, geralmente ingeridos de Data Lakes ou streaming.
Público-Alvo	Engenheiros de Dados, Desenvolvedores ETL.	Cientistas de Dados, Engenheiros de Dados, Engenheiros de ML.	Engenheiros de Dados, Analistas de Dados, Arquitetos de Dados, Cientistas de Dados.	Analistas de Dados, Engenheiros de Dados, Cientistas de Dados, Desenvolvedores de BI, Usuários de Negócios.	Engenheiros de Dados, Analistas de Dados, Desenvolvedores (para análise de telemetria).
Nível de Gerenciamento	PaaS	PaaS	PaaS	SaaS	PaaS
Diferencial Chave	Orquestração robusta e conectividade ampla.	Ambiente colaborativo e otimizado para Spark e ML.	Integração de Data Warehouse e Big Data Analytics em uma única plataforma.	Experiência unificada de ponta a ponta, centrada em dados (OneLake) e orientada a SaaS.	Ingestão e consulta de altíssima velocidade para dados de telemetria e séries temporais.

Azure SQL Edge

O Azure SQL Edge é um motor de banco de dados relacional otimizado, projetado para ser executado em dispositivos de Internet das Coisas (IoT) e na borda da rede (edge computing). Possui footprint (pegada de memória) para operar com eficiência em ambientes com recursos limitados.