Index für Developer und DBA

Arne Hattendorf

Zusammenfassung

Eine Welt, zwei unterschiedliche Sichten. Indizierung einmal aus der Sicht eines DBA und einmal aus der Sicht eines Developers.

DBA:

Der Entwickler der Software hat das Projekt vor einiger Zeit verlassen. Die Nachfolger verwalten mehr oder weniger nur, ohne die Software wirklich zu warten. Dann wird plötzlich festgestellt, dass "die Datenbank langsam" ist. Was nun? Beim letzten Kunden waren alle Advisories lizenziert und man hatte die Tuning-Funktionalitäten zur Verfügung. Selbst tunen kann sich die alte Möhre auch nicht.

Die Datenbank an sich scheint in Ordnung zu sein, es gibt (wie immer) keine Hinweise, dass Netzwerk oder SAN ein Flaschenhals sein können. Da gab es doch mal in grauer Vorzeit ein Statspack im Oracle. Das kann ich zwar einrichten und finde auch einige SQL, die problematisch sind. Jetzt muss ein Index her ... aber ...

Developer:

Du hast eine Anwendung übernommen. Einige SQL-Abfragen laufen nicht gerade hoch performant, obwohl auf manchen Tabellen Indices liegen. Neue Indices müssen her, aber welche?

Du entwickelst eine neue Anwendung. Dir ist schon klar, dass man Primärschlüssel indiziert. Wann kümmerst du dich um die anderen Indices, die nötig sind, und wie stellst du fest, was du überhaupt benötigst?

Wenn man nicht das Glück hatte, ein einem Projekt Erfahrungen mit Indizierung zu machen, steht man manchmal ziemlich allein da. Im Studium lernt man nichts oder wenig darüber, in den meisten SQL-Schulungen fehlt die Zeit, das ausführlich zu behandeln.

Allgemein

Index, Bedeutung und Struktur

Indices sind weithin bekannt, egal ob zum Auffinden von Stichwörtern in Büchern oder zum Auffinden von Büchern in einer Bibliothek. Auch ein Telefonbuch ist eigentlich nichts anderes, als ein großer Index. In einer Datenbank ist er ein physikalisches Objekt.

Ein Index hat eine Baumstruktur, vereinfacht gesehen enthalten die Zweige die Werte der Schlüsselspalte, die Blätter zeigen auf die Datensätze in der Tabelle. Hat man einen zusammengesetzten Index (mehrere Schlüsselfelder), ist außerdem die Reihenfolge der Spalten zu beachten.

Funktionsweise von Indices

In erster Linie sorgt ein Index für einen schnellen Zugriff auf eine Zeile einer Tabelle. Zusätzlich kann er die Kosten einer Sortierung erheblich senken. Sinnvolle Nutzung von Indices wird erschwert, wenn Spalten mit vielen NULL oder wenigen unterschiedlichen Werten indiziert werden. Eine falsche Spaltenreihenfolge führt zu weniger Effizienz, genau wie das Indizieren zu langer oder zu vieler Spalten.

Index Scans im Execution Plan

Lässt man einen xplan erstellen, wird die Art des Indexzugriffs sichtbar. Unique und Range Scan sind effiziente Formen eines Indexzugriffs. Ein Skip Scan oder Full Scan sind Hinweise auf Verbesserungspotential. Fast Full Scan und Index Join sind Sonderfälle mit normalerweise hoher Effizienz.

DEV spezifisch

Wann indizieren

Schon bei der Architektur der Software müssen Indices berücksichtigt werden. Keys bekommen Indices, schon jetzt sollten typische Abfragen der Anwender bedacht werden. Während der Softwareentwicklung müssen ggf. ebenfalls Indices angelegt werden, sei es für neue Objekte oder um Abfragen zu beschleunigen. Indizierung wird immer sofort gemacht, nie auf „später irgendwann“.

Wie indizieren

Abgesehen von Schlüsselfeldern werden Indices angelegt um SQL Abfragen zu beschleunigen. Jedes neue SQL sollte zumindest kurz mit Execution Plan geprüft werden, der SQL Developer macht das einfach möglich. Auf Suchen der Anwender und SQL, die in Funktionen aufgerufen werden, sollte ein besonderes Augenmerk gelegt werden. Indices auf NUMBER und CHAR Spalten sind einfach, bei DATE muss überlegt werden, ob Datum (vor allem bei vorhandener Uhrzeit) oder Character (z. B. bei Monat und Jahr) indiziert wird.

Programmierung

Bei der Softwareentwicklung ist zu berücksichtigen, dass die Indices auch genutzt werden. Dazu sind einfache Regeln zu berücksichtigen, außerdem hilft das Überprüfen der Ausführungspläne.

DBA spezifisch

Tools

Generell ist aus verschiedenen Gründen zu empfehlen, Statspack oder AWR Snapshots einzurichten, bevor man auch Probleme stößt. Nützliche Tools kommen z. B. aus dem Enterprise Manager der SQL Access Advisor oder zum Erzeugen einer Realitätsnahen Last Real Application Testing (RAT). Nützlich sind natürlich auch die Tools, die Diagnostic und Tuning Pack bieten hilfreich. Vorsicht, die Features haben teils sehr hohe Lizenzkosten.

Info Views

Auf die Informationen der Snapshots kann auch direkt über die STATS$ bzw. DBA_HIST Views zugegriffen werden. DBA_INDEX bzw. DBA_IND stellen ebenfalls nützliche Informationen zur Verfügung.

Tipps und Tricks

Ein paar Tricks können sich als sehr nützlich erweisen, z. B. vor dem Laden größerer Mengen NOLOGGING zu aktivieren, oder beim Erstellen riesiger Indices zusätzlichen Platz im TEMP Tablespace zur Verfügung zu stellen. Dazu im Housekeeping ab und zu die Nutzung der Indices prüfen, so wie sicherzustellen, dass aktuelle Statistiken für jeden Index vorhanden sind.

Fazit

Ein Index ist kein Hexenwerk. Wenn man ungefähr weiß, wie er funktioniert, kann man Zugriffe auf Tabellen stark beschleunigen. Dabei sind ein paar einfache Dinge zu vermeiden, die einen Index nutzlos werden lassen.

In erster Linie ist der Developer für die Indizierung verantwortlich, die am besten schon beim Design umfassend berücksichtigt wird. Idealerweise wird sie im Laufe der Entwicklung und nach Lasttests ständig angepasst.

Dem DBA stehen umfangreiche Werkzeuge zur Verfügung, um Entwickler zu unterstützen oder selbst tätig zu werden, sobald die Software in Betrieb gegangen ist. Im Idealfall ergänzen sich Developer und DBA hervorragend.

Für eine vernünftige Zusammenarbeit ist erforderlich, dass Developer und DBA sich gegenseitig respektieren. Jeder Part hat seine speziellen Aufgaben, Probleme, Stärken und Schwächen, was beim Thema Indizierung besonders deutlich wird.