



addconf.ru

Application Developer Days
Конференция программистов

29-30 АПРЕЛЯ 2011. Санкт-Петербург



Особенности масштабирования систем планирования и управления поставками

Михаил Антонов

Magenta Development



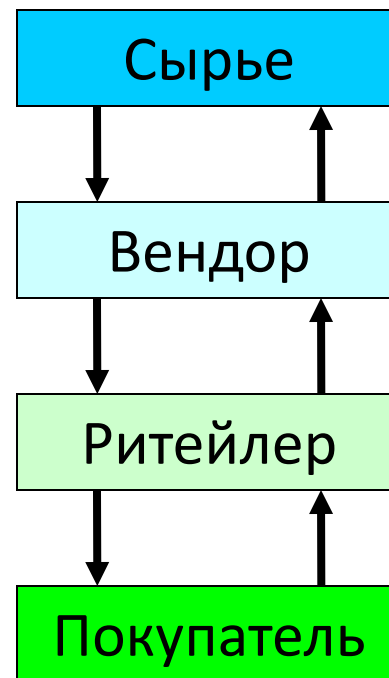
Масштабируемость нужна крупным:

- Интернет-проектам
 - highscalability.com
- А также банкам, биржам, интернет-провайдерам, системам планирования поставок и еще много где...
- ...но требования и подход везде разные



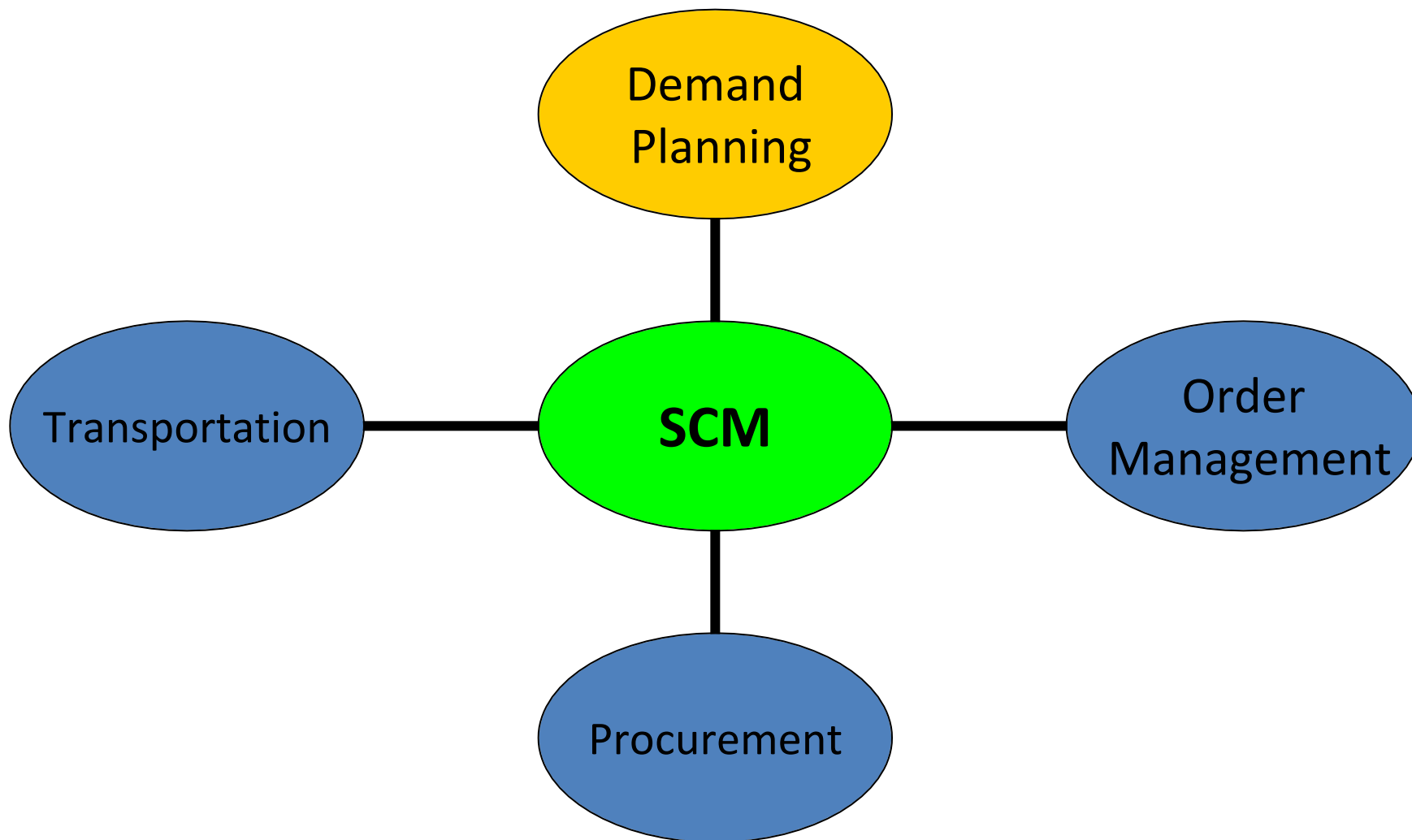
Как выглядит управление поставками (supply chain management, SCM)

- Сверху вниз – производится и поставляется товар
- Снизу вверх – «обратная связь», корректировка поставок





Из каких компонентов состоит?





Что делает ритейлер?

- Заказывает товары у вендоров
- Продает их покупателям в своих магазинах
- (В идеале) получает с этого прибыль
 - Оборот крупного ритейлера 50-100 млрд. \$ в год
 - Но чистая прибыль обычно 1-4% от оборота
 - Поэтому точность прогнозов продаж важна



Что нужно ритейлеру от SCM-системы?

- Получать данные о происходящем в его магазинах (продажи, скидки и т.п.)
- Накапливать информацию для анализа
- Делать **точные** прогнозы продаж на будущее
- Формировать и размещать заказы
- Иметь возможность наблюдать за происходящим / анализировать / вносить корректировки вручную



Какие трудности при разработке?

- Крупные компании медлительны
 - «быстро уточнить» или «попросить исправить на своей стороне» трудно
- Ограниченный стек технологий
 - Mainstream, поддержка, сертификация
- Точность прогнозирования
 - Проверять и тюнить нужно на живой системе
 - А ошибки там обходятся дорого
- Масштабирование...



Специфика масштабирования – пользователи

- Их мало (десятки, сотни), и они эксперты в предметной области
- Обычно они «тушат пожары»
 - Обнаруживать и тушить их нужно быстро
- Основываясь на данных
 - Максимально подробных
 - Агрегированных на различных уровнях
 - Удобно визуализированных



Специфика масштабирования – данные

- Их много, и их надо хранить и обрабатывать
 - 2 тысячи магазинов * 50 тысяч товаров * 500 дней
 - Это 50 миллиардов, но в реальности 5-10 миллиардов Points of Sales
 - А еще информация об Events, Inventory Profile etc
- Они приходят каждый день со всех магазинов
 - Их надо конвертировать, валидировать, загружать
 - Приходят в определенное время, но иногда задерживаются



Специфика масштабирования – данные, ч.2

- Их надо обрабатывать
 - Статистический (долгосрочный) прогноз
 - Эвристический (краткосрочный) прогноз
 - Создание заказов
- Время на обработку ограничено
 - Запускается после получения последних данных
 - Должна быть завершена до начала следующих фаз бизнес-процесса (упаковка, отгрузка товара со складов)
 - Обычное ограничение – 1-1.5 часа на все шаги обработки



Стек технологий

- Oracle 10g / 11g, RAC (RedHat Linux)
- Java 1.6, JBoss AS 4.2 (Windows 2003)
- Собственный Cache / Grid Manager
- JSP, Struts, Javascript



Собственно масштабирование

- Хранение массивных таблиц и индексов
- Быстрая загрузка интеграционных данных извне
- Оптимизация отдельных запросов
- Настройка и мониторинг БД «в целом»
- Оптимизация пользовательского интерфейса
- Оптимизация работы engines



Oracle Partitioning

- Разбиение таблиц и индексов на отдельные секции
- Которыми можно управлять индивидуально (add, drop, move, split и др.)
- Улучшает производительность (partition pruning, partition-wise joins, разнесение секций по нескольким физическим устройствам хранения)
- Прозрачно для SQL запросов (в теории)
- Экономически эффективное использование Storage devices



Примеры table partitioning

List partitioning

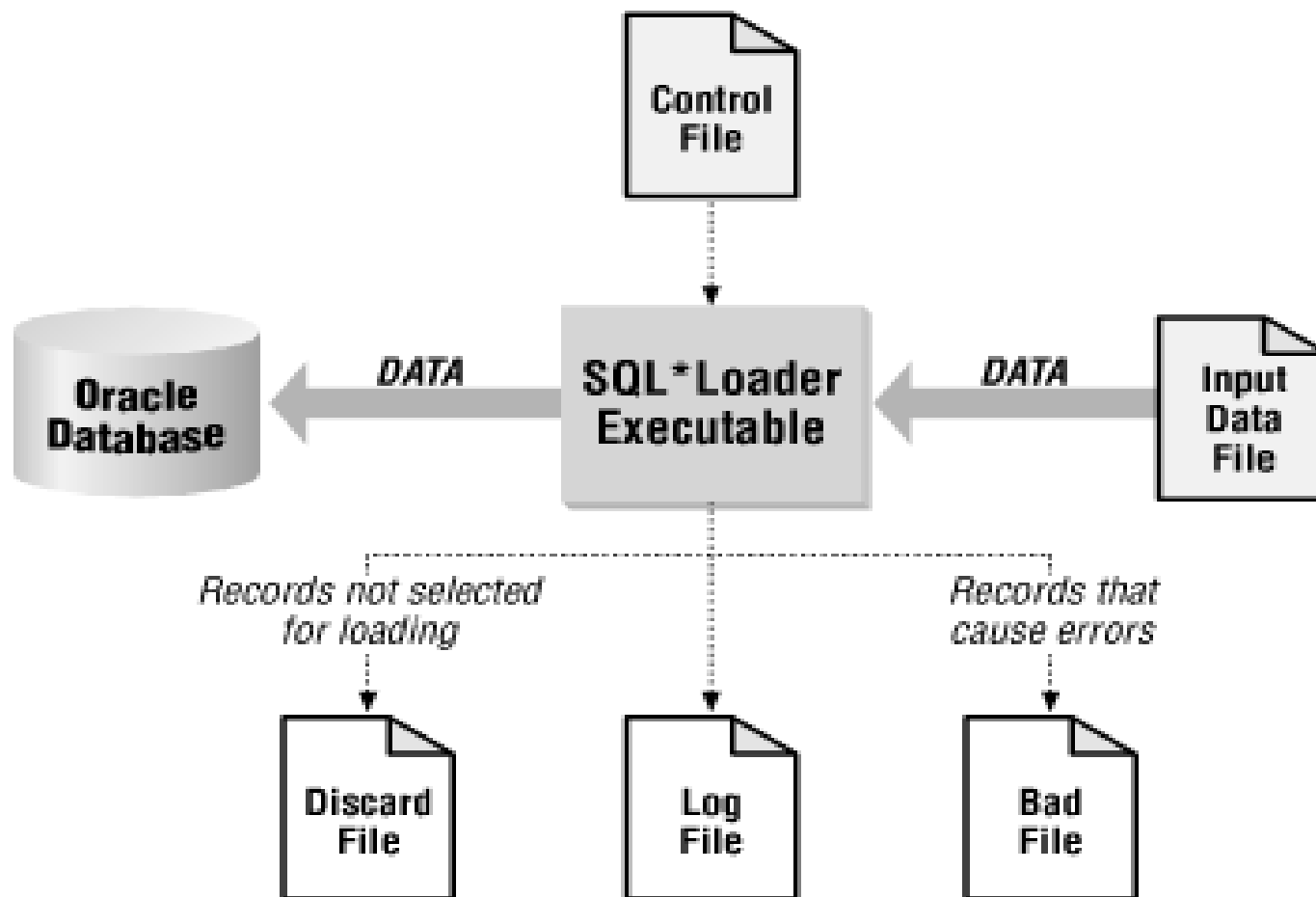
```
CREATE TABLE employers (  
    emp_no NUMBER PRIMARY KEY,  
    ename VARCHAR2(30),  
    deptno NUMBER)  
PARTITION BY LIST (deptno) (  
    PARTITION p10 VALUES (10),  
    PARTITION p20 VALUES (20,30));
```

Hash partitioning

```
CREATE TABLE employers  
    (id NUMBER,  
    name VARCHAR2 (60))  
PARTITION BY HASH (id)  
PARTITIONS 3  
STORE IN (tablespace tsname1, tsname2, tsname3);
```



Загрузка данных - Oracle SQL Loader





Пример использования SQL Loader

- **Control file**

load data

CHARACTERSET UTF8

infile 'c:\data\mydata.csv'

into table employer

fields terminated by "," optionally enclosed by ""

(empno, empname, sal, deptno)

- **Вызов**

sqlldr username@server/password control=loader.ctl



SQL Loader – ускорение загрузки

- Два режима загрузки - Conventional Load и Direct Path Load
- Отключение индексов и constraints, более редкие коммиты и data saves
- Увеличение размера буфера SQL Loader'а (чтение данных из файла более крупными блоками)
- Для Direct Path Load:
 - unrecoverable mode (отключение записи redo-логов)
 - параллельная загрузка (разбиение файла с данными на несколько более мелких, каждый из них загружается отдельным процессом SQL Loader)



2-х шаговая загрузка через SQL Loader

- Создаем временную таблицу той же структуры, что и CSV-файл, загружаем в неё CSV-file
- Разбиваем данные во временной таблице на группы
- Переносим данные из временной таблицы в основную
 - Используя UPDATE и / или INSERT
 - Создавая по отдельному task для каждой группы, и выполняя их параллельно
 - Заполняя нужные дополнительные столбцы (внешние ключи и др.)
 - Обновляя индексы
- Удаляем временную таблицу



External tables

- Хранятся не внутри tablespace, а как указатели на внешний flat file
- Позволяют обращаться к данным в flat file (CSV) как к таблице, используя SQL
- Являются дополнительной надстройкой над SQL Loader и работают через него



Жизненный цикл запроса

- Проверка синтаксиса
- Проверки обращений к объектам БД
- Трансформация запроса оптимизатором
- Оценка статистики, выбор способа выполнения
- Генерация Explain Plan
- Выполнение запроса (доступ к данным)



Explain plan

- Иерархичен
- Измеряется в costs – смысл зависит от модели оптимизации
- Требуется знания:
 - Базовых методов доступа к данным: FTS, index lookup, row id
 - Типов индексов – b-tree index, bitmap index
 - Методов доступа по индексу: unique scan, range scan...
 - Физические способы joins
- Оптимизатору можно давать подсказки (hints)
 - `SELECT /*+ HINT_NAME(params) */ ...`
 - `FIRST_ROWS(n)`, `ALL_ROWS`, `APPEND`.



Мониторинг БД – Oracle Grid Control

ORACLE Enterprise Manager 10g
Grid Control

Home Targets

Hosts | Databases | Middleware | Web Applications | Services | Systems | Groups | All Targets

Database Instance: KRGPSR >

Monitored SQL Executions

Active in last 12 hours

Status	Duration	SQL ID	Session	Parallel	Database Time	IO	Start	Ended
	2.3m	bhu4qw7dcm29	254		2.3m	25K	05:21:14 AM	
	1.0s	0jarcnr1r67ty	487		1.1s	73	05:16:38 AM	05:16:39 AM
	2.0s	afyl <code>select count(*) from store_order_schedule</code>			12.7s	431	05:16:31 AM	05:16:33 AM
	6.0s	gg7169jvqq6mz	487	8	28.3s	431	05:16:20 AM	05:16:26 AM
	1.0s	bhr0nchzp3hru	487	2	1.2s	99	05:16:08 AM	05:16:09 AM

[Home](#) | [Targets](#) | [Deployments](#) | [Alerts](#) | [Compliance](#) | [Jobs](#) | [Reports](#) | [Setup](#) | [Preferences](#) | [Help](#) | [Log](#)



Oracle Grid Control - продолжение

Monitored SQL Execution Details

Overview





SQL ID	bhu4qw7dcm29 ⓘ	Time		IO & Wait Statistics
Execution Started	Fri Apr 23 2010 05:21:14 AM	Duration	 1.3m	IO Count 
Last Refresh Time	Fri Apr 23 2010 05:21:14 AM	Database Time	 1.3m	Buffer Gets 
Execution ID	16777216	PL/SQL & Java	0.0s	Wait Activity % 0
Session	254			
Fetch Calls	0			

`select count(*) from store_order_schedule`

Details

 Plan Statistics  Activity

Plan Hash Value 405534056

Operation	Name	Estimated Rows	Cost	Timeline(78s)	Executions	Actual Rows	Memory	Temp	CPU Activ
SELECT STATEMENT					1				
SORT AGGREGATE		1			1				
PARTITION LIST ALL		12M	2843		1	5732K			
PARTITION HASH ALL		12M	2843		4	5732K			
INDEX FULL SCAN	STORE_ORDER_SCHEDULE_PK	12M	2843		1727	5732K			



Transient Kernel Profiler (tkprof)

- `alter session set sql_trace=true;`
- `alter session set timed_statistics=true;`
- <Выполнение любых запросов>
- Форматирование файла трассировки
- Анализ результатов по выполненным запросам:
Количество разборов запроса, процент попаданий в кеш, процессорное время, количество прочитанных блоков, число извлеченных строк и др.
- Не забыть выключить трассировку!



Оптимизация UI

- Пре-агрегирование данных (materialized views, вспомогательные таблицы с ручным обновлением)
- Вынос туда повторяющихся «тяжелых» частей запросов
- Регулярное обновление / очистка вспомогательных таблиц / mviews
- Принудительная фильтрация в UI как мера предосторожности



Оптимизация engines

- Выделение цепочки engines для обработки данных
- Выделение групп данных, которые могут рассчитываться независимо друг от друга. Группировка должна быть одинаковой по всем массивным таблицам. Пример:
 - таблицы: Forecasts, Orders, Sales, Alerts
 - subnet: Distribution center (DC) - Item
- Создание параллельно выполняющихся задач для обработки каждой группы каждым engine
 - Горизонтально масштабируемо
 - Более мелкие и управляемые транзакции
 - Отслеживание прогресса, предсказуемое время выполнения



Выводы о масштабируемости

- Узкое место – I/O на серверах БД
- Хотя многие операции проще и быстрее выполнять внутри СУБД, Oracle масштабировать дороже, чем Java/JBoss
- Оптимально выглядит 4-6 Java-серверов и 1-2 сервера Oracle



Вопросы?

Михаил Антонов

Magenta Technology

antonov.michael@magenta-technology.ru

twitter.com/Zorkus