

Oracle 9i

Für Einsteiger

Teil I

The Oracle logo is displayed on a black rectangular background. The word "ORACLE" is written in a bold, red, sans-serif font with a registered trademark symbol (®) at the end. Below it, the tagline "SOFTWARE POWERS THE INTERNET" is written in a white, bold, sans-serif font.

ORACLE®
SOFTWARE POWERS THE INTERNET

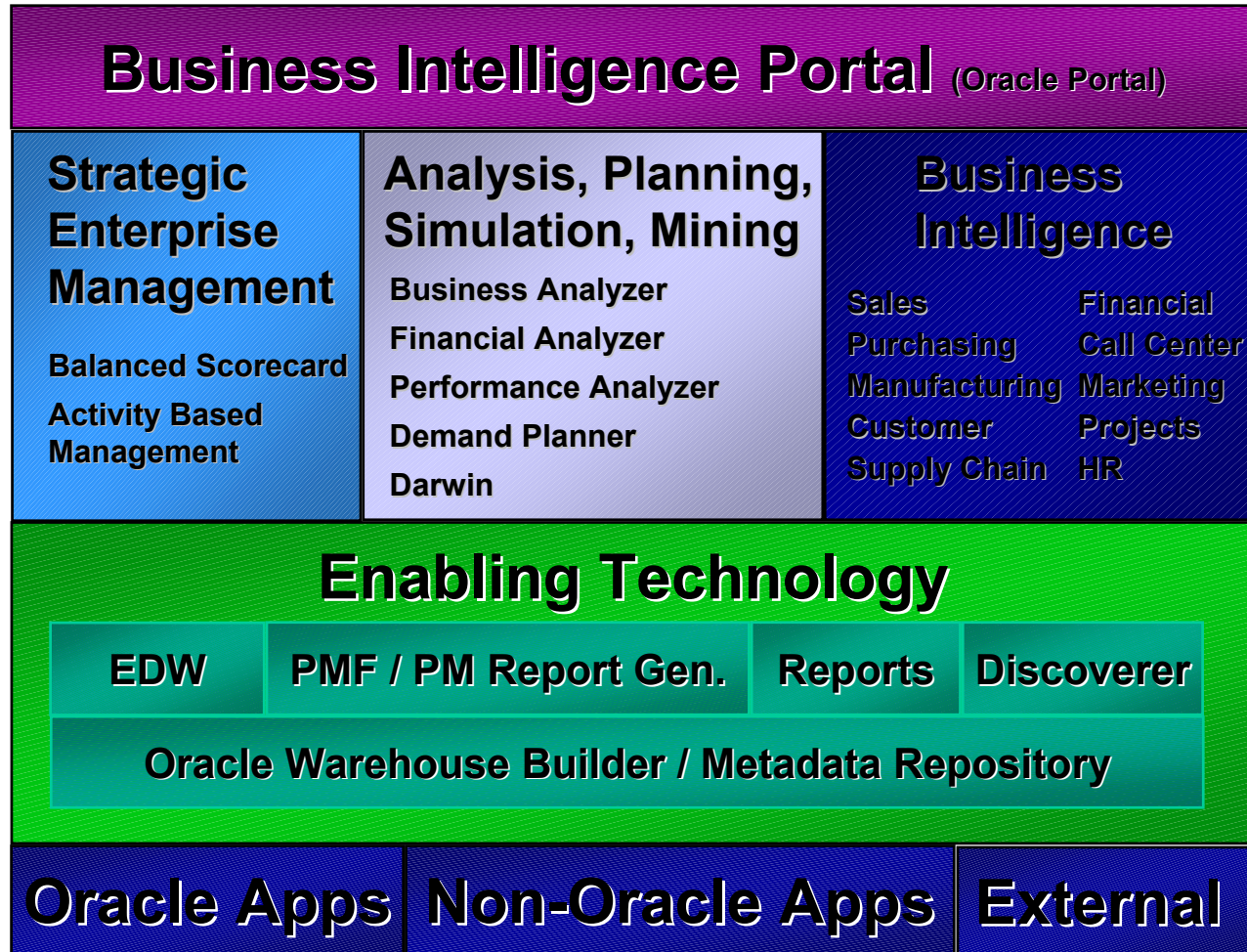
E-business Intelligence Suite 11i

Personalized View

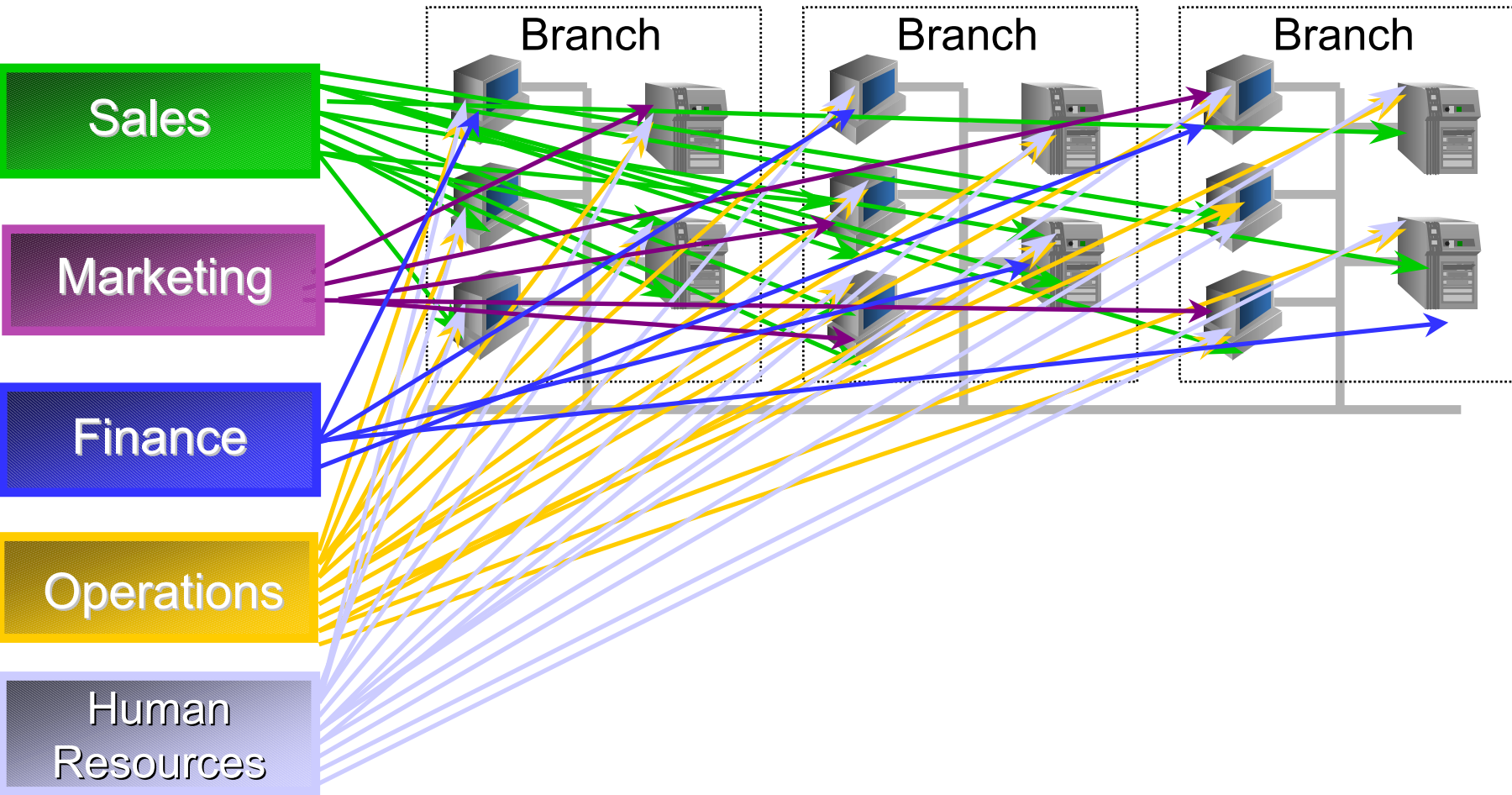
**Analysis
Align Strategy...**

**Accessibility
Availability
Accuracy**

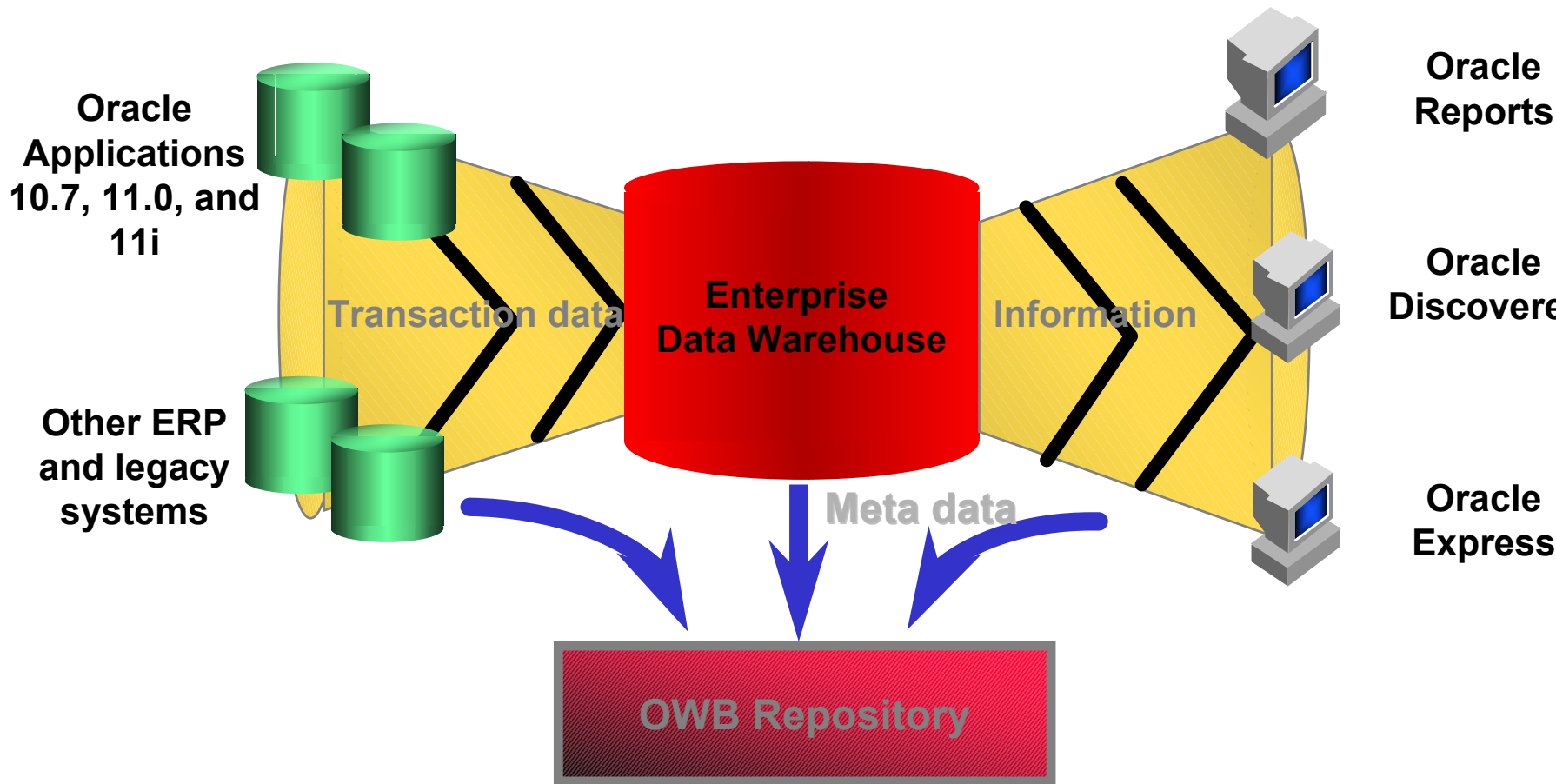
Multiple Sources



Data Mgmt. Is the Central Solution Theme



Business Intelligence System Architecture Overview With EDW



Der Oracle Server

- Terminologie
- Die serverarchitektur
- Hintergrund-supportprozesse
- Init.Ora
- Die Steuerdatei
- Redo logs
- Datenbank-Datendateien
- Rollback-Segment / Undo Tablespace
- Wichtige Speicherstrukturen
 - Daten-Cache
 - Library-Cache
 - Locks und Latches

Der Oracle Server

- Der Oracle Server bildet das Rückgrad der Oracle-Technologie.
- Themen dieses Moduls:
 - Einführung in die Architektur des Servers
 - Die Supportprozesse im Hintergrund
 - Datenbankinstanz
 - Initialisierungsparameterdatei
 - Datendateien
 - Redo Logs
 - Steuerdateien
 - Rollback Segmente und deren Funktion
 - Highlights der Hauptspeicherstruktur
 - Locks und Latches

Der Oracle Server - Terminologie

- Oracle-Instanz
 - Besteht aus mehreren gemeinsam genutzten Speicherprozessen, die den Mechanismus für den Zugriff auf einen Set von Oracle Datenbankdateien zur Verfügung stellen.
- Init.ora
 - Initialisierungsparameterdatei, die beim Starten von Oracle gelesen wird.
- Applikationsserver
 - Programm auf einem Rechner in einem verteilten Netzwerk, das einem Applikationsprogramm die Geschäftslogik zur Verfügung stellt.
- Checkpoint Aktivität
 - Oracle Aufgaben wie die Datensynchronisation und Integritätsprüfung (die automatisch ablaufen)

Der Oracle Server - Serverarchitektur

- **Init.ora**
 - Enthält Einträge zur Festlegung der Laufzeitumgebung
 - Die Werte dieser Datei steuern den Umfang des Hauptspeichers und vieles mehr
- **Shared Memory**
 - Teil des Hauptspeichers, der beim Start einer Oracle Datenbank reserviert wird (SGA: System Global Area)
- **Datenbankdateien**
 - Datendateien
 - Steuerdateien
 - Redo-Logdateien
- **Support Prozesse**
 - Eigentliche Arbeiter, die beim Starten angelegt werden.

Der Oracle Server - Supportprozesse

- Database Writer (dbwr)
 - Schreibt Inhalte der Datenbank von den Puffern auf die Laufwerke
 - “Dirty” Puffer:
 - Puffer, deren Daten geändert wurden
 - Standardmässig gibt es einen dbw0, maximal 9 dbwr-Prozesse
 - Falls ein Benutzerprozess den Zugriff auf Daten benötigt, die sich nicht im Puffer-Cache befinden, stellt der dbwr sicher, dass ein freier Puffer zur Verfügung gestellt wird.
 - dbwr ist der einzige Prozess, der an die Datenbank schreibt
 - Man nennt den Prozess auch “Kommunikationsdirektor”

Der Oracle Server - Supportprozesse

- Process Monitor (pmon)
 - Ist für die Aufräumarbeiten zuständig, falls eine Benutzersitzung abbricht und das Aufräumen nicht mehr selber durchführen kann.
 - Ressourcen einer abgebrochenen Sitzung werden durch pmon freigegeben.
 - pmon wird auch als „Geheimagent“ bezeichnet
- System Monitor (smon)
 - Beim Starten stellt smon sicher, dass alle Datenbankdateien konsistent sind und führt gegebenenfalls Recovery durch.
 - smon prüft in regelmässigen Abständen, ob irgendwelche Aufgaben zu erledigen sind.
 - Die Arbeit dieser Prozesse sichert den reibungslosen Betrieb einer Oracle Instanz
 - Man könnte den Prozess als „Rechnungsprüfer“ bezeichnen.

Der Oracle Server - Supportprozesse

- Log Writer (lgwr)
 - Kümmt sich um die Verwaltung der Redo-Log Puffer, den Transaktionslogs von Oracle.
 - Werden Transaktionen initiiert und schliesslich festgeschrieben (gesichert) oder über ein Rollback zurückgesetzt, wird für diese Aktivitäten ein Datensatz an die Redo-Logdateien geschrieben
 - Der Log Writer entspricht dem „Kassenwart“, der alle Transaktionen protokolliert.
- Checkpoint (ckpt)
 - Aktualisiert alle Datendateien der Datenbank
 - Der Datenbank Writer ist der einzige Prozess, der Daten in die Datenbankdateien schreibt, während ckpt sicherstellt, dass die Datendateien nach Beendigung synchron sind.
 - Dieser Prozess entspricht dem „Fahrdienstleister“, der alles synchronisiert.

Der Oracle Server - Supportprozesse

- Recoverer (reco)
 - Ist in erster Linie für die Auflösung gescheiterter Transaktionen in einer verteilten Oracle Umgebung zuständig.
- Archiver (arc0)
 - Speichert automatisch (falls die Media Recovery eingeschaltet ist), automatisch Kopien der Redo Logs an einen vom DBA vorgegebenen Standort
 - Dies entspricht einem Eintrag im init.ora
 - **log_archive_start = true**

Der Oracle Server – init.ora

```
# INIT file for my oracle instance  
#  
audit_trail = none  
background_dump_dest = /d0/oraclehome/product/oracle9.0.1/...  
compatible = 9.0.0.0  
control_files = (/d1/oradata/beg9/redo1/ora_xlhtbycl.ctl,  
/d1/oradata/beg9/redo2/ora_xlhtbykt.ctl)  
core_dump_dest = /d0/oraclehome/product/oracle9.0.1/...  
cursor_sharing = force  
db_block_size = 16384  
db_cache_size = 50m  
db_create_file_dest = /d1/oradata/beg9  
db_create_online_log_dest_1 = /d1/oradata/beg9/redo1  
db_create_online_log_dest_2 = /d1/oradata/beg9/redo2  
db_name = beg9
```

Der Oracle Server – init.ora

```
# INIT file for my oracle instance  
#  
...  
distributed_transactions = 10  
hash_area_size = 8388608  
java_pool_size = 20971520  
large_pool_size = 614400  
log_archive_start = true  
log_checkpoint_interval = 0  
log_checkpoint_timeout = 0  
log_checkpoints_to_alert = true  
max_dump_file_size = 10000  
open_cursors = 200
```

Der Oracle Server – init.ora

- Standorteinträge
 - **control_files = (/d1/oradata/myoracle/redo1/ora_xlhtbycl.ctl,**
 - **/d1/oradata/myoracle/redo2/ora_xlhtbykt.ctl)**
 - **core_dump_dest = /d0/oraclehome/product/oracle9.0.1/. . .**
 - **db_create_file_dest = /d1/oradata/myoracle**
 - **db_create_online_log_dest_1 = /d1/oradata/myoracle/redo1**
 - **db_create_online_log_dest_2 = /d1/oradata/myoracle/redo2**
- Bei falschen Angaben erfolgt eine Fehlermeldung
 - **SQL> startup**
 - **ORA-0044: Background process „LGWR“ failed while starting**

Der Oracle Server – init.ora

- Limitierende Einträge
 - Ressourcenbeschränkungen
 - **open_cursors = 200**
 - Speicherreservierungen
 - **db_cache_size = 50m**
 - m steht für Megabytes, g für Gigabytes, k für Kilobytes sonst Bytes
- Einträge für Funktionen
 - Funktionsumfang : false, true, partial, full
 - **oracle_trace_enabled**
 - Ebene, auf der eine Funktion zu aktivieren ist: z.B. compatible
 - **compatible = 9.0.0.0**
 - **compatible = 9.0.1.0**

Der Oracle Server – init.ora

- Ändern von Parametern
 - Bei früheren Versionen des Oracle Servers liessen sich die Parameter nur ändern, indem man
 - die Init.ora bearbeitete,
 - die Datenbank herunterfuhr
 - und neu startete
 - Seit Version 7 kann man viele Parameter bei laufender Datenbank ändern.
 - **SQL> desc v\$parameter**
 - Zeigt den Aufbau der Systemview auf die Parameter
 - Die Spalte `issys_modifiable` enthält die Werte
 - **false, deferred, immediate**
 - » **SQL> select issys_modifiable, name from v\$parameter where issys_modifiable = „IMMEDIATE“ and name like „db%“;**

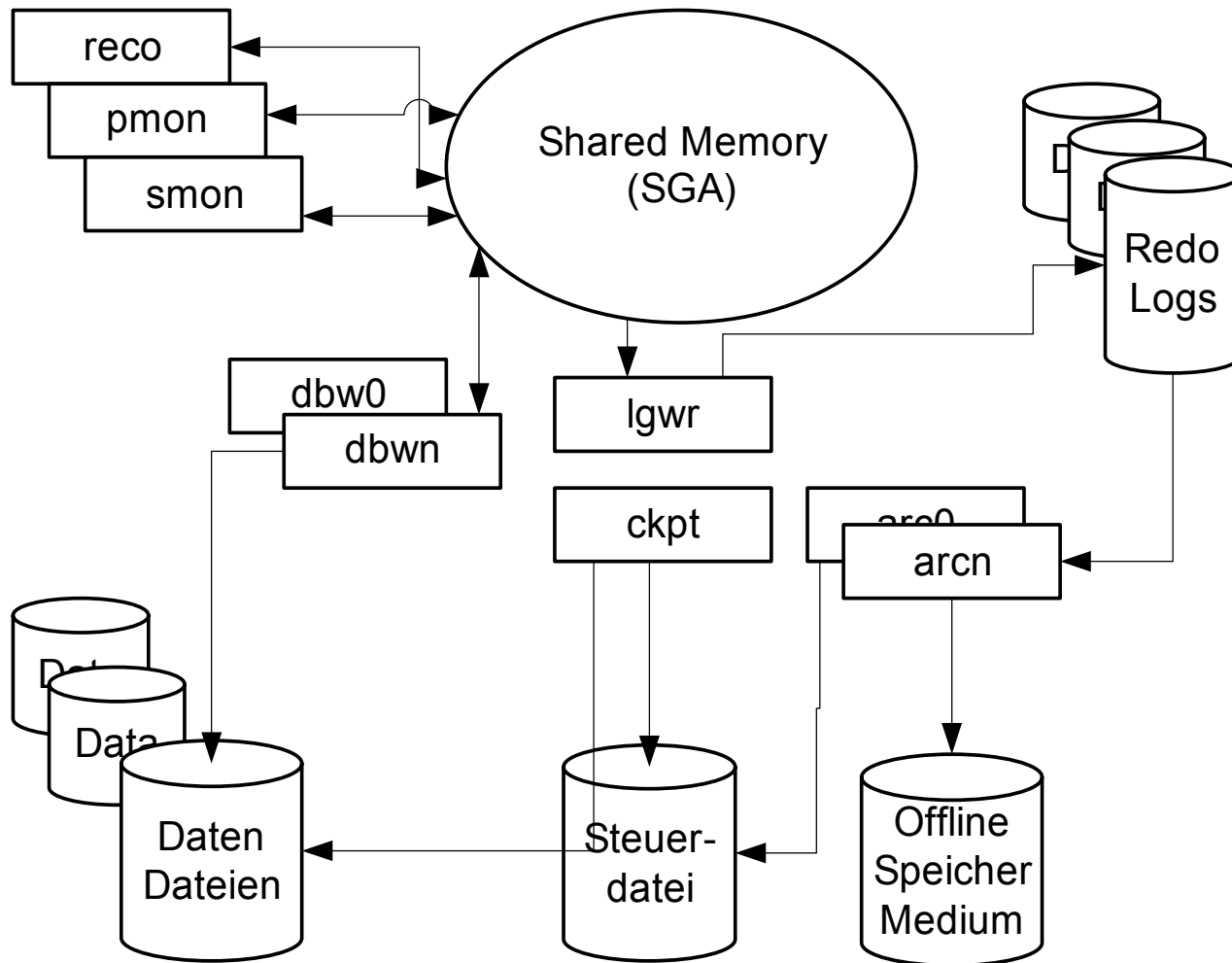
Der Oracle Server – Control File (Steuerdatei)

- Wird beim Anlegen der DB eingerichtet, eine pro Instanz
- Infos:
 - Name der Datenbank
 - Pfadnamen für alle Datenbank- und Redo-Log-Dateien
 - Datum und Uhrzeit des Anlegens
 - Aktuelle Log-Sequenznummer
(Oracle 9i alloziert diese beim Start einer neuen Logdatei)
 - Checkpoint Informationen
- Infos
 - SQL> desc v\$controlfile
beschreibt den Aufbau der Data Dictionary View

Der Oracle Server – Redo Logs

- Sind das Herz des Oracle Servers!
 - Ab Version 6
 - Ab Version 7 als Redo Loggruppen
- Abfrage
 - **SQL> desc v\$logfile**
liefert eine Beschreibung der View
 - **SQL> col member for a40**
SQL> select * from v\$logfile order by 1,2;
zeigt alle Redo Log Dateien an
- Funktion
 - Beim Interagieren mit der Datenbank wird bei einem commit oder rollback ein Datensatz über die Aktivitäten an die Redo Logs geschrieben.

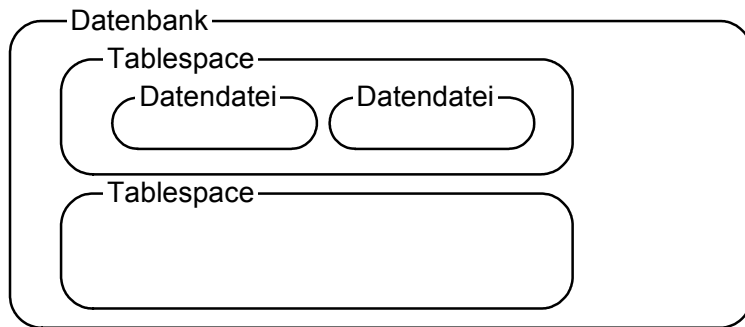
Der Oracle Server – Redo & Background Prozesse



Der Oracle Server – Datenbank-Datendateien

- Daten werden in Datenbank-Datendateien gespeichert
- Systemdateien legt man mit einem Werkzeug an
 - SQL*PLUS oder OEM
- Datendateien werden über ein Tablespace in eine Instanz eingebunden
 - **create tablespace g1**
datafile ,/d0/oradata/myoracle/kurs/g101.dbf‘ size 2000m,
,/d1/oradata/myoracle/kurs/g102.dbf‘ size 2000m
blocksize 2048
default storage ...;
 - Achtung:
 - **db_2k_cache** in init.ora muss mit dem obigen **blocksize** Parameter übereinstimmen.

Der Oracle Server – Datenbank-Datendateien



- **create database**
 - Legt den Tablespace SYSTEM an
- **Starten von Administrations-Skripten**
 - Legen den Rest des Data Dictionnaires an

Rollback-Segment/Undo Tablespace

Status	Aktivität	Undo Speicher	Daten im Speicher	Ansicht für andere
1	Vor Transaktion 1	leer	940.--	940.--
2	update konto set kontostand = kontostand + 12000;	940.--	12940.--	940.--
3	rollback;	leer	940.--	940.--
4	update konto set kontostand = kontostand + 1200;	940.--	2140.--	940.--
5	commit;	leer	2140.--	2140.--

Der Oracle Server – Wichtige Speicherstrukturen

- Der Daten-Cache
 - Ein Teil der SGA wird für Daten reserviert, die aus den Datenbankdateien gelesen werden und/oder in diese zu schreiben sind.
 - Die Grösse des Daten-Cache wird über diverse init.ora Parameter bestimmt
 - [Oracle9] **db_cache_size**
 - [Oracle8] **db_block_size** und **db_block_buffer**
 - Oracle verwendet einen “Least Recently Used” Algorithmus, um Daten aus dem Daten-Cache (Hauptspeicher) auszulagern
 - Datenmanipulationen werden mit den Daten aus dem Daten-Cache ausgeführt.

Der Oracle Server – Wichtige Speicherstrukturen

- Der Library-Cache
 - Nimmt alle SQL und PL/SQL Anweisungen auf
 - Die Grösse wird über den Parameter **shared_pool_size** kontrolliert
 - Abarbeitungsfolge (in Oracle 9)
 - Anweisungen werden an Oracle übergeben
 - Es wird ein Hash-Wert generiert und die Anweisung im Library Cache abgelegt.
 - Oracle vergleicht den Hash-Wert mit Hash-Werten in einer Tabelle im Cache.
 - Ist eine Übereinstimmung vorhanden, wird die neue SQL Anweisung verworfen und stattdessen jene aus dem Cache ausgeführt.
 - Ist keine Übereinstimmung vorhanden, wird die neue SQL-Anweisung in den Library Cache gelegt und der Hash-Wert in die Hash-Tabelle eingetragen.

Der Oracle Server – Wichtige Speicherstrukturen

- Locks und Latches (Sperrungen und Schutz der Daten)
 - Locks
 - Stellen das korrekte Interagieren verschiedener Sitzungen sicher (**select** ist immer erlaubt; mutieren muss koordiniert werden: **rollback** macht eine Änderung rückgängig; **commit** bestätigt eine Änderung)
 - Sperrmodi in Oracle
 - Exclusive
 - » Es sind keine anderen Zugriffe erlaubt (Änderungen)
 - » Gleichzeitige Zugriffsversuche werden in eine Warteschlange eingetragen
 - Shared
 - » Gleichzeitiger Lesezugriff ist erlaubt (Lesen)

Der Oracle Server – Wichtige Speicherstrukturen

- Locks und Latches (Sperrungen und Schutz der Daten)
 - Latches
 - Sind Mechanismen auf Zeilenebene
 - Sie werden im Millisekundenbereich erworben und freigegeben.
 - Latches benötigt man bei der Arbeit mit gemeinsamen Strukturen in der SGA.
 - *Willing to Wait Latches*
 - Ein Latch kann nicht unmittelbar erworben werden
 - Es werden weitere Anforderungen verschickt (bis es klappt)
 - » Beispiel: Latches im Library Cache
 - *No Wait Latches*
 - Werden storniert, falls das Latch nicht unmittelbar erhältlich ist.
 - » Beispiel: Redo Log Kopien
 - Der DBA braucht sich in der Regel nicht um Latches zu kümmern!

Der Oracle Server – Einrichten einer DB

- Wählen Sie ORACLE_HOME (z.B. /d1/oracle/kurs)
- Installieren Sie die SW mit dem Installer
- Wählen Sie für die neue DB einen Oracle System Identifier ORACLE_SID (z.B.: myoracle)
- Legen Sie init.ora an
 - Hinterlegen Sie diese unter ORACLE_HOME/dbs (database in NT)
 - Im obigen Beispiel : initmyoracle.ini
- Einrichten des DB Skripte
- Ausführen der Data Dictionary Skripte
- Einrichten des Tablespaces für die Daten
- Einrichten der Benutzer



Oracle für Einsteiger

Sind Sie neu im Geschäft?

Sind Sie neu im Geschäft?

- Datenbankobjekte
 - Terminologie
 - Tabellen
 - Views
 - Indizes
 - Funktionen, Prozeduren
- SQL*Plus
 - Data Definition Language (DDL)
 - Data Manipulation Language (DML)
- PL/SQL – das kleine Einmaleins
 - PL/SQL: Die Programmiersprache von Oracle
- DBA – das kleine Einmaleins
 - Was ist ...?

Sind Sie neu im Geschäft?

Datenbankobjekte

Datenbankobjekte

- Tabellen
- Views
- Indizes
- Trigger
- Synonyme
- Sequenzen
- Rollen
- Funktionen, Prozeduren

Datenbankobjekte - Terminologie

- Data Dictionary
 - Umfasst alle Informationen darüber, wie Daten in der Datenbank gespeichert sind, wo sie liegen und wie die Datenbank damit arbeiten kann
- DDL (data definition language)
 - Alle SQL Anweisungen, die mit **create, grant, revoke, drop** beginnen. Mit diesen Anweisungen können DBObjekte gelöscht werden.
- DML (data manipulation language)
 - Sind SQL Anweisungen, die mit **select, insert, update** beginnen
- Integritätsregeln
 - Sind die Geschäftsregeln in der DB (Bestellung nur falls Kunde)
- DB-Trigger
 - Programme, in der DB, die durch Events aufgerufen werden. (z.B. beim Einfügen eines Datensatzes in eine Tabelle).

Datenbankobjekte - Tabellen

- Datenbankobjekt, das alle Daten aufnimmt (Benutzer&System)
- Tabelle = <Spalte 1><Spalte2>...<Spalte k>
- Spalte i besteht aus Daten eines bestimmten Datentyps (**int, varchar2, date, ...**)
- Falls immer Werte eingegeben werden müssen
 - **not null**
- Sonst
 - **null**

Datenbankobjekte - Tabellen

- Beispiel

- SQL> **CREATE TABLE HAUSTIER (**
 HT_ID INTEGER PRIMARY KEY,
 HT_ART VARCAHR(20) NOT NULL
 HT_NAME VARCHAR2(20) ,
 HT_GESCHL CHAR(1)
 CHECK(HT_GESCHL IN ('M','W'))
)
- SQL> /
Table created.

Datenbankobjekte - Tabellen

- create table as
 - Anlegen einer Tabelle basierend auf einer anderen Tabelle
 - **SQL>CREATE TABLE informatik_studenten AS
select *
from studenten
where studenten.abteilung="Informatik"**
SQL>/
 - **SQL>select * from informatik_studenten;**

zeigt alle Informatikstudenten.

Datenbankobjekte - Views

- Sind Datensichten aus einer oder mehreren Basistabellen
- Besteht aus einer gespeicherten SQL Anweisung (Abfrage)
- Einsatzgebiet
 - Sie wollen bestimmten Mitarbeitern nicht alle Daten sichtbar machen.
- Beispiel
 - **SQL> CREATE VIEW abteilungsleiter AS**
SELECT m.vorname, m.name, a.abteilung
FROM mitarbeiter m, abteilung a
WHERE m.abt_nr = a.abt_nr
AND m.status = “Abteilungsleiter”;

SQL>/
 - **SQL>SELECT * FROM abteilungsleiter;**
Rem zeigt alle Abteilungsleiter

Datenbankobjekte – Indices

- Indices beschleunigen den Zugriff und die Suche
- Eindeutige und Nichteindeutige Indices
 - **SQL>CREATE UNIQUE INDEX ui_abteilungsnr
ON abteilung (abteilungsnr);
SQL>/**
- Primärschlüssel
 - Kann beim Kreieren einer Tabelle angegeben werden
 - **SQL> ALTER TABLE abteilung
ADD CONSTRAINT pk_abteilungsname
PRIMARY KEY (abteilungsname)
SQL>/**
 - Achtung:
 - Primärschlüssel lassen sich nicht mehr problemlos löschen!
- Faustregel
 - Die indexierte Abfrage ist schneller als die direkte, falls der Index weniger als 5% der Daten der Tabelle umfasst.

Datenbankobjekte – Trigger

- Trigger sind Programme, die in der Datenbank gespeichert sind und beim Auftreten eines bestimmten Ereignisses ausgeführt werden (in PL/SQL, Java oder C)
- Oracle gestattet die Definition von Triggern für
 - Insert, update, delete
- Beispiel

```
– CREATE OR REPLACE TRIGGER trg_ht_art
  BEFORE
  DELETE OR INSERT OR UPDATE
  ON haustiere
  FOR EACH ROW

BEGIN
  if INSERTION then
  insert into log (log_id, log_table, ...) values
  (.....)
  ...
END trg_ht_art;
```

Datenbankobjekte – Synonyme

- Ein Synonym ist ein alternativer Name für eine Tabelle, View, Sequenz oder ein Programm
- Gründe für Synonyme
 - Verbergen des tatsächlichen Namens für den Eigentümer der Datenbankobjekte
 - Verbergen des tatsächlichen Standardorts des DB-Objekts
 - Anbieten eines einfachen Namens für ein Objekt.
- Man unterscheidet
 - Private (für einen Benutzer) und öffentliche (DB weit) Synonyme

Datenbankobjekte – Synonyme

- Beispiel
 - SQL>Rem privates Synonym für den Benutzer meier
 - SQL>**CREATE SYNONYM gehaltsstufe
for meier.gehaltsstufen**

SQL>/

SQL>Rem öffentliches Synonym

SQL>**CREATE SYNONYM gehaelter
for firmengehaltstabelle**

SQL>/

SQL>**SELECT * FROM gehaelter;**

Datenbankobjekte – Sequenzen

- Generieren Zahlensequenzen, beispielsweise für Schlüssel
- Beispiel
 - **SQL>CREATE SEQUENCE auftrags_nr
START WITH 1000 INCREMENT BY 1
MINVALUE 1
CACHE 1000 NOCYCLE NOORDER
SQL>/**
- Sequenzen werden im Memory gespeichert, stehen also sehr schnell zur Verfügung.
- Beim Herunterfahren des DBMS gehen die Sequenzen verloren.

Datenbankobjekte – Funktionen, Prozeduren, Packages

- Funktionen

- Beispiel

- **SQL>CREATE FUNCTION neuerPreis(in_artNr IN INTEGER)
RETURN NUMBER
IS preisNeu NUMBER(11,2);
BEGIN
 SELECT preis+0.5*preis
 INTO preisNeu
 FROM artikel
 WHERE artikel.artNr = in_artNr;
RETURN(preisNeu);
END;**

SQL>/

- Aufruf

SQL>var x number;

SQL>exec :x := neuerPreis(127865)

SQL>print x;

...(Anzeige des neuen Preises)

Datenbankobjekte – Funktionen, Prozeduren, Packages

- Prozeduren

- Beispiel

- **SQL>CREATE OR REPLACE PROCEDURE NEUER_PREIS(
 in_artNr IN INTEGER,
 out_neuerPreis OUT NUMBER)
AS
...
BEGIN
 SELECT preis+0.5*preis
 INTO out_neuerPreis
 FROM artikel
 WHERE artikel.artNr = in_artNr;
EXCEPTION
 WHEN ...
END;
SQL>/**

Datenbankobjekte – Funktionen, Prozeduren, Packages

- Package

- Fasst mehrere Prozeduren und Funktionen zusammen

- Beispiel

- **SQL>CREATE OR REPLACE PACKAGE meineUtils AS
PROCEDURE komplexesFraktal(in_z, out_z);
FUNCTION mandelbrot(in_z, in_x) RETURN NUMBER;
END meineUtils;**

SQL>/

Datenbankobjekte – Andere Datenbankobjekte

- Sie können beispielsweise direkt Verzeichnisse kreieren
- Beispiel
 - **SQL>CREATE OR REPLACE DIRECTORY meinPuffer
AS '/d0/oraclehome/temp';**

Sind Sie neu im Geschäft?

SQL*PLUS

SQL*PLUS

- Zugang zu SQL*PLUS
- DML und DDL: wo liegen die Unterschiede?
- SELECT (Grundformat)
- Formattieren von selektierten Daten
- SET Befehle
- Verknüpfen von Tabellen

SQL*PLUS - Zugang

- Starten von SQL*PLUS
- \$sqlplus oder über ein Symbol
 - SQL>....
Enter user-name:meinLogin
Enter password: *****
SQL>...
- Beenden der SQL*Plus Sitzung
 - SQL>**EXIT**

SQL*PLUS – Data Definition Language (DDL)

- Zu DDL gehören alle Befehle, die Datenbankobjekte wie Tabellen, Views, ... anlegen oder löschen.
 - ALTER PROCEDURE
 - ALTER TABLE
 - ANALYZE
 - ALTER TABLE ADD CONSTRAINT
 - CREATE TABLE
 - CREATE INDEX
 - DROP INDEX
 - DROP TABLE
 - GRANT
 - REVOKE

SQL*PLUS – Data Definition Language (DDL)

- Oracle Datentypen
 - **char(size)** Zeichendaten mit fixer Länge size
 - **varchar2(size)** Zeichendaten variabler Länge (max)
 - **number(l, d)** numerisch, Länge l, d Dezimalstellen
 - **blob** binary large object (bis 4GB binär)
 - **raw(size)** wie varchar2 für binäre Daten
 - **date** Datumsangabe
 - **long** bis zu 2 GB Zeichendaten

SQL*PLUS – Data Definition Language (DDL)

- describe
 - Beispiel
 - SQL>**desc meineTabelle**
 - Zeigt die Details meiner Tabelle meineTabelle an
- not null
 - Falls null als Datenwert zugelassen wird, kann der Eintrag in dieses Tabellen-Attribut vollständig fehlen
 - Falls Sie not null angeben, muss ein Wert eingegeben werden, also ein praktischer Test für viele Anwendungen!

SQL*PLUS – Data Manipulation Language (DML)

- Dazu gehören alle SQL-Anweisungen, die mit `select`, `insert`, `update` oder `delete` beginnen.
 - SELECT
 - Selektiert Daten aus der Datenbank
 - INSERT
 - Lädt Daten in die Datenbank
 - UPDATE
 - Ändert Daten in der Tabelle
 - DELETE
 - Löscht Daten aus einer Tabelle

SQL*PLUS – Data Manipulation Language (DML)

- INSERT
- Beispiel

Version 1

- **SQL>insert into meineTabelle('Peter', 'Wyss','01-Feb-78');**

Version 2

- **SQL>insert into cd_sammlung (cd_index, cd_kurzbezeichnung)
values ('1002', 'The Wall');**
- In der zweiten Variante werden nur bestimmte Daten der Tabelle eingefügt.

SQL*PLUS – Data Manipulation Language (DML)

- SELECT
- Aufbau
 - **select** (obligatorisch)
welche Informationen möchte ich
 - **from** (obligatorisch)
wo befinden sich die Informationen (Oracle Tabelle)
 - **where** (optional)
spezielle Bedingungen, welche die Daten einschränken
 - **group by** (optional)
Gruppierung der Daten (nach sort-Kriterien)
 - **order by** (optional)
Sortierung der Daten

SQL*PLUS – Data Manipulation Language (DML)

- SELECT
- Formattieren der Ausgabe einer select-Anweisung in SQL*Plus
 - **column** definiert die Spalte, die formatiert werden soll
 - **format** <Maske> Grösse der Ausgabe
 - **heading** ‚String‘ Spaltenüberschrift, ‚|‘ entspricht ‚neue Zeile‘
 - **wrap/trunc** Zeilenumbruch-Verhalten
- Beispiel
 - SQL>**column cd_kurzbezeichnung format a10 heading ‚Werk‘**

SQL*PLUS – Data Manipulation Language (DML)

- SELECT
- Formattieren der Ausgabe einer select-Anweisung

– Formatmasken:

Formatzeichen	Beispiel	Beschreibung
• A	format a10 trunc	Anzeigebreite 10 Zeichen
• 9	format 99999	Anzeigebreite Anzahl 9 Zahlen
• 0	format 0999	zeigt führende Nullen an
• ...		

SQL*PLUS – Data Manipulation Language (DML)

- **SELECT**
 - ORDER BY
 - **SQL> select cd_index, cd_kurzbezeichnung from cd_sammlung
order by cd_index desc, gruppe, cd_kurzbezeichnung;**
 - **SQL>select cd_index, cd_kurzbezeichnung from cd_sammlung
order by 1 desc, gruppe, 2;**
Rem sortieren über nummerierte Positionen.

SQL*PLUS – Data Manipulation Language (DML)

- Die SQL*Plus Umgebung
 - **set pagesize / linesize**
legt die Seitengrösse fest (Länge, Breite)
 - **spool <Ausgabedatei>**
leitet die Ausgabe in die <Ausgabedatei> um
 - **set termout on/off**
schaltet die Ausgabe ins Terminal aus
 - **show all**
zeigt die aktuellen Einstellungen an

SQL*PLUS – Data Manipulation Language (DML)

- Verknüpfen von Tabellen

- Diese erfolgt in der where-Klausel

- Beispiel

- **SQL>select cd.gruppe, fa.kaufdatum
from cd_sammlung cd, buchhaltung fa
where cd.cd_index = fa.kurzbezeichnung
order by 1,2;**

- Dabei haben wir für die Tabellen jeweils alternative Namen definiert

<Tabelle>	<alternativer Name>
cd_sammlung	cd
Buchhaltung	fa

SQL*PLUS – Data Manipulation Language (DML)

- break on
 - Gruppenbildung geschieht in SQL*Plus mit der break Anweisung
 - SQL>**break on gruppe**
SQL>**select cd.cd_index, cd.gruppe, cd_kurzbezeichnung**
from cd_sammlung cd
order by 1,2;
- compute sum of <Feld> on <Break>
 - SQL>**compute sum of salary on state_name**
SQL>**break on state_name skip 1**
SQL>**select st.state_name, nh.lname, nh.hiredate, nh.salary**
from newhire nh, state st
where st.state_cd = nh.state_cd
order by 1,2;

Sind Sie neu im Geschäft?

PL/SQL – das kleine Einmaleins

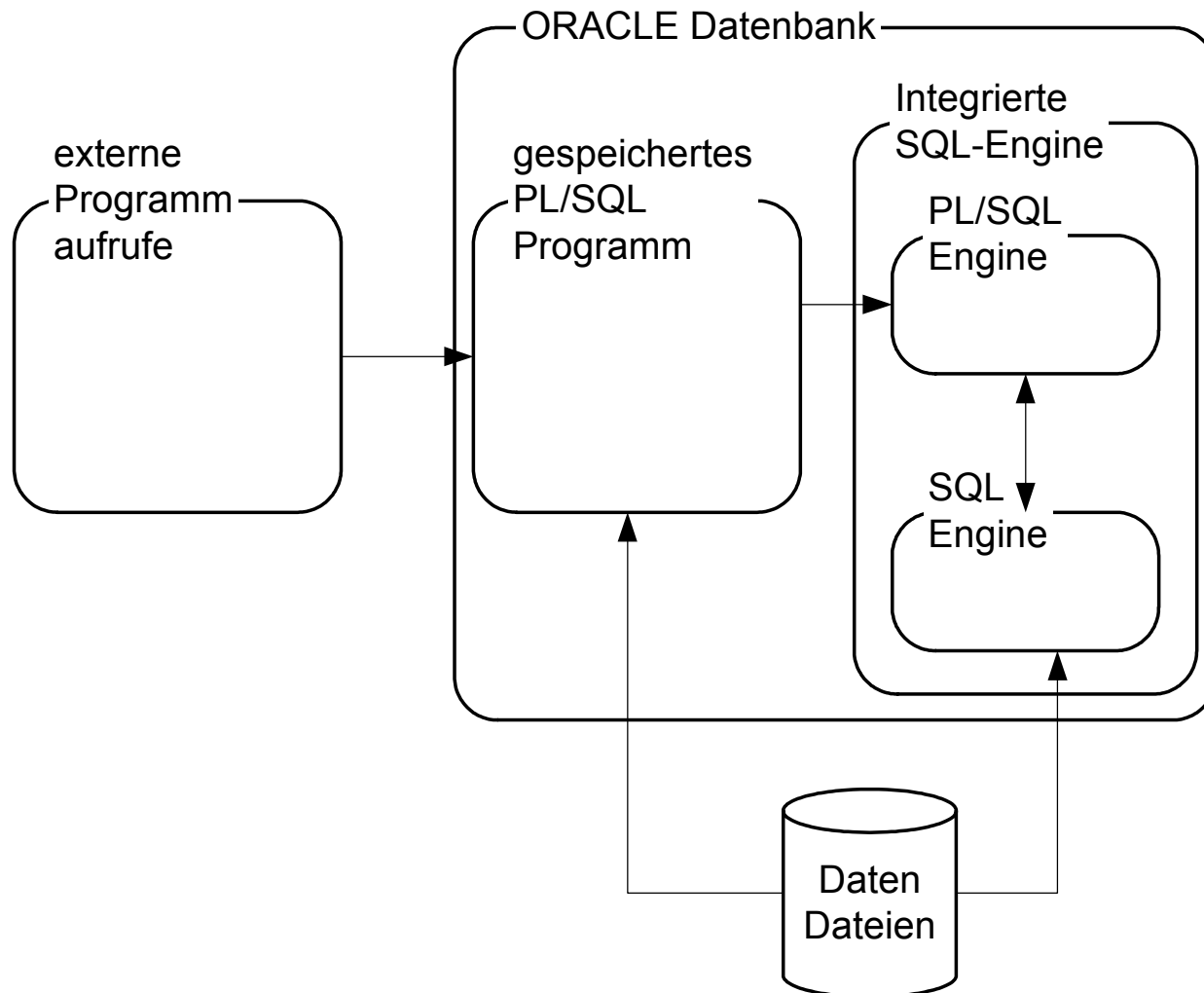
PL/SQL

- Überblick über die Struktur von PL/SQL
- Die Komponenten von PL/SQL
- Übersetzen und ausführen
- Die Blockstruktur
- Variablen und Datentypen
- Ausführungssteuerung
- Schleifen
- SQL in PL/SQL
- Cursor
- Ausnahmen / Exceptions
- Gespeicherte Funktionen und Prozeduren
- Das Debugging

PL/SQL - Terminologie

- **Declare Section**
 - Bereich, in dem die Programmvariablen definiert werden
- **Executable Section**
 - Bereich, in dem definiert wird, was das Programm macht
- **Ausnahmen / Exceptions**
 - System- und benutzerseitige Mechanismen zur Fehlerbeseitigung
- **Datentyp**
 - Klasse von Informationen oder Daten
- **[anonymer] PL/SQL Block**
 - Ein Set von Befehlen / das PL/SQL Programm
- **Gespeichertes Objekt**
 - PL/SQL Codesegment, das in der DB gespeichert wird

PL/SQL – Die Programmiersprache von ORACLE



PL/SQL – Die Programmiersprache von ORACLE

- PL/SQL wird in zahlreichen Oracle Produkten eingesetzt
 - Oracle Server
 - Oracle Forms
 - Oracle Report
 - Oracle Warehouse Builder
 - Oracle Applications
 - Oracle Portal
- PL/SQL Code lässt sich aus folgenden Entwicklungs-Umgebungen aufrufen
 - SQL*Plus
 - Oracle Enterprise Manager
 - Oracle Precompiler (wie Pro*C)
 - Oracle Call Interface (OCI)
 - Service Manager
 - Oracle 9i Applikations Server
 - Java Virtual Machine

PL/SQL – Der Zeichensatz

- Unterstützte Zeichen
 - Gross- und Klein-Buchstaben
 - Alle Ziffern zwischen 0 und 9
 - Symbole ()+-*/<>=!:;:'@%#&_ | {\$?[]
- Arithmetische und relationale Operatoren
 - ** Exponent
 - *, / Multiplikation, Division
 - +, -, || Subtraktion, Addition, Verkettung
 - Relationale Operatoren
 - = gleich
 - <> oder != ungleich
 - > grösser
 - < kleiner
 - >= grösser oder gleich

PL/SQL – Struktur

- Grundsätzlicher Aufbau
 - **DECLARE**
 - Variablen**
 - **BEGIN**
 - Programm**
 - **END;**
- Kommentare
 - **/* Anfang bis
Ende */**
 - **-- bis Zeilenende**

PL/SQL – Variablen

- Datentypen (die häufigsten)
 - **varchar2, number, date, boolean**
- Schema
 - Deklaration
variablenName variablenTyp := variablenWert;
 - Normale Anweisung
variablenName := variablenWert;
- Beispiel
 - **declare** /* deklarationsteil */
l_counter number := 0;
l_today := sysdate;
l_name := varchar2(50);
begin /* ausführungsteil */
l_name='Esther';
...
end;

PL/SQL – Variablen

- **varchar2**
 - Alphanumerischer Datentyp mit variabler Länge, max 32,767 Bytes
 - Deklarationschema
 - **variableVarChar varchar2(maxLength); /* maxLength >0 */**
 - Deklarationsbeispiel
 - **L_name varchar2(30) := 'Abraham';**
- **number**
 - Numerischer Datentyp
 - Deklarationschema
 - **variableNumber(length, decimalPoint); /*length:0..32;*/**
 - Deklarationsbeispiel
 - **N_variable number(12,2);**

PL/SQL – Variablen

- **date**
 - Datumswerte mit fixer Länge
 - Deklarationschema
 - **variableDate date;**
 - Deklarationsbeispiel
 - **d_Datum date := '11-SEP-99';**
- **boolean**
 - Wahrheitswert (**true** oder **false**)
 - Deklarationschema
 - **variableBoolean boolean;**
 - Deklarationsbeispiel
 - **b_variable := 12>2;**

PL/SQL – Steuerstrukturen

- If-Logikstrukturen

- **if...then**

```
IF l_date > '11-APR-03' THEN
    l_salary := l_salary * 2;
END IF;
```

```
IF l_date > '11-SEP-99' THEN
    IF l_flight <> 'AA' THEN
        l_status := 'okay';
    END IF;
END IF;
```

PL/SQL – Steuerstrukturen

- If-then-elsif Logikstrukturen

- **if...then ... elsif**

```
IF l_date > '11-APR-03' THEN
  l_salary := l_salary * 2;
ELSIF l_date > '10-JAN-02' THEN
  l_salary := l_salary * 1.2;
END IF;
```

Falls die erste Bedingung nicht zutrifft, wird eine weitere Bedingung geprüft (elsif).

Das Konstrukt erinnert an eine case (Auswahl) Anweisung...

PL/SQL – Steuerstrukturen

- case Ausdrücke

- beschreibt eine Auswahl aus mehreren Alternativen

- **CASE** variable

- WHEN** ausdruck1 **THEN** wert1

- WHEN** ausdruck2 **THEN** wert2

- ELSE** wert5

- END;**

- Beispiel

- **val:= CASE** city

- WHEN** ‚ZUERICH‘ **THEN** ‚Grasshoppers‘

- WHEN** ‚BERN‘ **THEN** ‚Young Boys‘

- ELSE** ‚keine Mannschaft‘

- END;**

- Die Lesbarkeit im Vergleich zu **if then elsif ...** ist wesentlich besser!

PL/SQL – Steuerstrukturen

- Schleifen

- Syntax

- LOOP**

- ausführbare Anweisungen;

- END LOOP;**

- mit **EXIT** kann die Schleife vorzeitig verlassen werden.

- Beispiel

- **LOOP**

- IF l_gehalt > 10000 THEN EXIT;**

- ELSE l_gehalt = l_gehalt*1.12;**

- END IF;**

- END LOOP;**

PL/SQL – Steuerstrukturen

- WHILE-Schleifen

- Syntax

- WHILE** Bedingung
ausführbare Anweisungen;
END LOOP;

- mit **EXIT** kann die Schleife vorzeitig verlassen werden.

- Beispiel

- **WHILE**

- IF total_einkaeufe > 10000 THEN EXIT;**
ELSE total_einkaeufe := total_einkaeufe + artikel_preis;
END IF;
END LOOP;

PL/SQL – Steuerstrukturen

- FOR-Schleifen

- Syntax

- FOR** Laufvariable **IN** Laufvariablen-Werte
LOOP

- ausführbare Anweisungen;

- END LOOP;**

mit **EXIT** kann die Schleife vorzeitig verlassen werden.

- Beispiel

- **FOR** l_counter **IN** 1..10 /* zwei Punkte */
LOOP

- IF** total_einkaeufe > 10000 **THEN** **EXIT;**

- ELSE** total_einkaeufe := total_einkaeufe + artikel_preis;

- END IF;**

- END LOOP;**

PL/SQL – SQL in PL/SQL-Programmen

- SQL kann direkt in PL/SQL eingesetzt werden
- Beispiel

– **declare**

l_emp_count number;

i number;

begin

select count(*) into l_emp_count from employees;

FOR i IN 1..l_emp_count LOOP

dbms_output.put_line(,Mitarbeiter ,||i);

PL/SQL – Cursor

- Das Beispiel zum Einsatz von SQL in PL/SQL zeigt, wie SQL in PL/SQL eingebunden werden kann
- Neues Schlüsselwort
 - **into** **select count(*) from ... into l_...;**
 - Erlaubt sind beliebige SQL Anweisungen (INSERT, UPDATE,...)
 - SQL öffnet dabei einen sogn. *impliziten Cursor*, einen Kanal in die Datenbank, implizit, weil der Benutzer nichts programmiert.
 - Implizite Cursor sind langsam!

PL/SQL – Cursor

- Explizite Cursor
 - declare
 - l_empl_count** number;
 - i** number;
 - CURSOR get_employee_data IS select count(*) from employees;**
 - begin**
 - OPEN get_employee_data;**
 - FETCH get_employee_data INTO l_empl_count;**
 - FOR i IN 1.. l_empl_count LOOP**
 - dbms_output.put_line(,Employee ,|| i);**
 - END LOOP;**
 - CLOSE get_employee_data;**
 - end;**

PL/SQL – Ausnahmebehandlung

- Sobald bei der Ausführung eines PL/SQL Programms Fehler auftreten, prüft das Programm ob eine Exception Section vorhanden ist und für den entsprechenden Fehler eine Fehlerbehandlung hinterlegt wurde.
- Häufige Exceptions in PL/SQL
 - **no_data_found** die select Anweisung liefert keine Daten
 - **too_many_rows** eine Anweisung liefert mehrere Zeilen
 - **value_error** inkompatible Datentypen
- Beispiel
 - **declare ...**
 - begin ...**
 - EXCEPTION**
 - WHEN no_data_found THEN**
 - raise_application_error(-20052,‘Sorry keine Daten‘);**
 - WHEN others THEN ...**
 - end;**

PL/SQL – Gespeicherte Prozeduren & Fktn

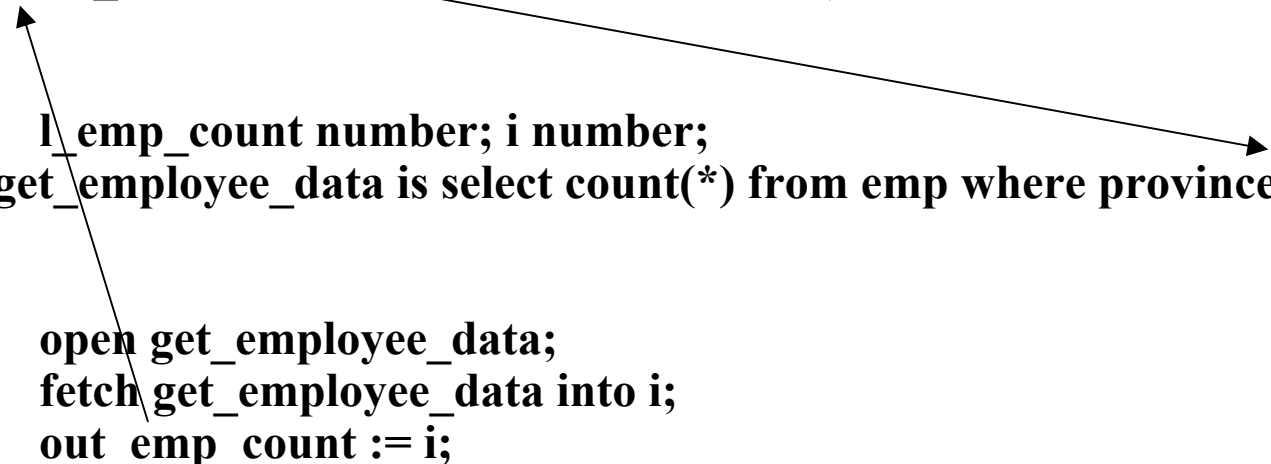
- Mit CREATE OR REPLACE PROCEDURE kann eine Prozedur permanent in der DB abgespeichert werden (DB Objekt Procedure).
 - **CREATE OR REPLACE PROCEDURE LIST_EMPLOYEES IS
BEGIN
 declare l_emp_count number; i number;
 cursor get_employee_data is select emp_name, salary from emp;
 begin
 for c1_rec in get_employee_data
 loop
 dbms_output.put_line(...');
 end loop;
 end;
END;**
- Test
 - SQL>execute list_employees

PL/SQL – Gespeicherte Prozeduren & Fktn

- Das Gleiche mit Parametern

- **CREATE OR REPLACE PROCEDURE EMPLOYEES_COUNT**
(IN_PR IN VARCHAR2,
OUT_EMP_COUNT OUT NUMBER)

```
IS  
BEGIN  
  declare l_emp_count number; i number;  
  cursor get_employee_data is select count(*) from emp where province=  
IN_PR;  
  begin  
    open get_employee_data;  
    fetch get_employee_data into i;  
    out_emp_count := i;  
  end;  
END;
```



- Test

- SQL>variable prov_count number;
SQL>execute employees_count('ONTARIO', :prov_count);
SQL>print -- zeigt die Variablen an

PL/SQL – Gespeicherte Prozeduren & Fktn

- Funktionen

- Beispiel

- **CREATE OR REPLACE FUNCTION COUNT_ALL_EMP
RETURN number
IS
BEGIN
 declare i number;
 cursor get_employee_data IS select count(*) from emp;
 BEGIN
 open get_employee_data;
 fetch get_employee_data INTO i;
 RETURN i;
 END;
END;**

- Test

- **SQL>select count_all_emp() from dual;**

PL/SQL – Debugging

- Analog zu anderen Programmiersprachen kann man die Daten zeilenweise ausgeben.
- In PL/SQL kann man dazu mit
 - SQL>**set serveroutput on**
oder
SQL>**set serveroutput on size 1000000**einschalten oder ausschalten
 - SQL>**set serveroutput off**
- Die Daten können zeilenweise ausgegeben werden, sofern der serveroutput auf on steht
 - SQL>**... dbms_output.put_line(...');**

Sind Sie neu im Geschäft?

DBA – das kleine Einmaleins

DBA – das kleine Einmaleins

- Was ist eine Datenbank?
- Was ist eine Oracle-Instanz?
- Die System Global Area
- Wie man die Datenbank startet
- Wie man die Datenbank herunterfährt
- Der Oracle-Tablespace
- Rollback-Segmente
- Redo Loop
- Steuerdateien
- Benutzerkonten (Accounts)

DBA - Terminologie

- Data Dictionary
 - Wird von der Datenbank verwaltet und enthält alle Informationen darüber, wie die Daten in der Datenbank gespeichert werden, wo sie liegen und wie man damit arbeiten kann.
- Extent
 - Eine aufeinanderfolgende Ansammlung von Oracle-Blöcken
- Block
 - Ist in einer Oracle-Datenbank die kleinste Speichereinheit (2KB)
- System Global Area (SGA)
 - Ist ein Bereich im Hauptspeicher, der für Datenbankinformationen genutzt wird (alle Benutzer greifen darauf zu)
- Instanz
 - Bei jedem Start einer DB wird eine SGA reserviert und die Oracle Hintergrundprozesse gestartet. Die Kombination nennt man Instanz.

DBA – Was ist eine Oracle-Instanz?

- Die System Global Area (SGA)
 - Beim Start einer Oracle-Instanz wird die SGA reserviert und die Oracle Hintergrundprozesse gestartet.
 - Die Kombination (SGA+Hintergrundprozesse) bezeichnet man als Instanz.
 - **SQL>show sga**
zeigt die Speicherreservierungsgrösse an (total, fix, variable...)
 - SGA umfasst
 - Datenpuffer
 - Redo-Log-Puffer
 - Shared Pool

DBA – Was ist eine Oracle-Instanz?

- Datenbankpuffer – ihr Daten-Cache
 - Die zuletzt (und oft mehr als einmal benutzte) Daten werden im Puffer zwischengespeichert werden.
- Redo-Logpuffer – ihr Transaktions-Cache
 - Der Redo-Puffer enthält alle Änderungen, die in der Datenbank vorgenommen wurden. Danach werden die Daten in das Redo-Log eingetragen.
- Oracle-Hintergrundprozesse
 - smon : System Monitor
 - pmon : Prozess Monitor
 - dbwr : Database Writer
 - lgwr : Log Writer
 - ckpt : Checkpoint

DBA – Was ist eine Oracle-Instanz?

– Oracle-Hintergrundprozesse

- smon : System Monitor
ist eine Instanz, die beim Neustart einer gescheiterten Instanz Recovery ausführt (z.B. wegen Stromausfall)
- pmon : Prozess Monitor
führt die Recovery für gescheiterte Benutzerprozesse durch
- dbwr : Database Writer
schreibt geänderte Blöcke aus dem Datenpuffer-Cache an die Datendateien.
- lgwr : Log Writer
schreibt Blöcke aus den Log Writer-Puffercaches an die Redo Logs
- ckpt : Checkpoint
ist der Punkt, an dem alle geänderten Datenbankpuffer in der SGA wieder in die Datendateien geschrieben wird.

DBA – Was ist eine Oracle-Instanz?

- startup open
 - Bevor Sie auf die Datenbank zugreifen können, müssen Sie diese starten (und damit auch die Hintergrundprozesse aktivieren)
 - SQL>**connect / as sysdba;**
SQL>**startup open test pfile=/d0/oraclehome/test/pfile/init.ora;**

startet die Datenbank Test mit der dazugehörigen init.ora Datei
- Init.ora
 - Enthält Parameter, welche die Ressourcen der DB beschreiben
 - SQL>**show parameters;**

DBA – Was ist eine Oracle-Instanz?

- startup nomount

- Beim ersten Starten der DB existiert diese ja noch nicht.
Die Hintergrundprozesse müssen aber auch so gestartet werden können.

```
SQL>connect / as sysdba;
```

```
SQL>startup nomount pfile=/d0/oraclehome/test/pfile/init.ora
```

```
SQL>CREATE DATABASE „TEST“
```

```
LOGFILE GROUP (./d0/oradata/redo01.log) SIZE 500K REUSE,
```

```
DATAFILE ./d1/oradata/test/system_001.dbf
```

```
SIZE 250M REUSE
```

```
MAXLOGFILES 20
```

```
CONTROLFILE REUSE
```

```
MAXDATAFILES 500
```

DBA – Was ist eine Oracle-Instanz?

- shutdown
 - SQL>**shutdown**
normales Herunterfahren von Oracle
 - SQL>**shutdown immediate**
diese Version wartet nicht bis alle Benutzerprozesse abgeschlossen sind
 - SQL>**shutdown abort**
wartet nicht bis die Benutzerprozesse und Systemprozesse abgeschlossen sind.

DBA – der Oracle-Tablespace

- Create Tablespace – extent management dictionary
 - Standardmässig legt Oracle einen Tablespace SYSTEM an, mit der Klausel **extent management dictionary**.
- Was ist ein Extent?
 - Ein Extent ist eine aufeinanderfolgende Sammlung von Oracle-Blöcken.
 - Ein Oracle-Block ist die kleinste physische Platzeinheit (init.ora Parameter)
 - **CREATE TABLESPACE user02
DATAFILE ,/d0/oraclehome/test/files/user02_001.dbf'
SIZE 5M REUSE
EXTENT MANAGEMENT DICTIONNARY
DEFAULT STORAGE (INITIALIZE 25K NEXT 25K
MINEXTENTS 5
MAXEXTENTS 100
PCTINCREASE 0);**

DBA – der Oracle-Tablespace

- Create Tablespace – extent management dictionary
 - *create tablespace – extent management local autoallocate*
 - **CREATE TABLESPACE user03**
DATAFILE ,/d0/oraclehome/test/files/user03_001.dbf‘
SIZE 5M REUSE
EXTENT MANAGEMENT LOCAL AUTOALLOCATE;
 - In diesem Beispiel werden die extents lokal im Tablespace verwaltet.

DBA – der Oracle-Tablespace

- Create Tablespace – extent management dictionary
 - *create tablespace – extent management local uniform size*
 - **CREATE TABLESPACE user04**
DATAFILE ,/d0/oraclehome/test/files/user04_001.dbf‘
SIZE 5M REUSE
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 128K;
 - Anfangsgrösse des Tablespaces : 128 KB
 - Erweiterungsschritte : 128 KB

DBA – der Oracle-Tablespace

- Create Tablespace – extent management dictionary
 - *create undo tablespace*
 - **CREATE UNDO TABLESPACE user05
DATAFILE ,/d0/oraclehome/test/files/user05_001.dbf‘
SIZE 2M REUSE
AUTOEXTENT ON;**
 - Wird in der Regel mit dem CREATE DATABASE Befehl verwendet.
 - Das UNDO- Segment ermöglicht es, eine Transaktion rückgängig zu machen.

DBA – der Oracle-Tablespace

- Create Tablespace – extent management dictionary
 - *alter tablespace add data file*
 - **ALTER TABLESPACE user04
ADD DATAFILE ,/d0/oraclehome/test/files/user04_002.dbf'
SIZE 2M REUSE
AUTOEXTENT ON MAXSIZE 10M;**
 - Legt eine weitere Datenbankdatei an.
 - Bei Bedarf wird weiterer Platz schrittweise (1MB) bis max 10MB hinzugefügt.
 - Ohne Wachstum:
**ALTER DATABASE
DATAFILE ,/d0/oraclehome/test/files/rbs_002.dbf'
AUTOEXTENT OFF;**
 - *alter tablespace offline*
 - **ALTER TABLESPACE user04 OFFLINE; //offline setzen
ALTER TABLESPACE user04 ONLINE; //online setzen**

DBA – der Oracle-Tablespace

- Einen Tablespace löschen
 - **SQL>DROP TABLESPACE user02 INCLUDING CONTENTS CASCADE CONSTRAINTS;**

löscht den Inhalt und den Tablespace (neu in 9i)
 - **SQL>DROP TABLESPACE user03 INCLUDING CONTENTS AND DATAFILES;**

löscht die physische Datendatei zusammen mit dem Tablespace
 - **SQL>DROP TABLESPACE user04;**

löscht den Tablespace (klassischer Befehl).

DBA – Rollback-Segmente

- Rollback-Segmente sind Segmente in der DB, die ein „Before“-Image oder eine „Undo“-Kopie eines Datenblocks vor dessen Änderungen enthält (am besten befinden sich diese in sep. Tablespaces ‚...*rb*s...‘)..
- Transaktionen nutzen entweder Rollback-Segmente oder Undo-Tblspace.
- *create rollback segment*
 - **SQL>CREATE ROLLBACK SEGMENT R77 TABLESPACE rbs STORAGE (INITIAL 250K next 250K minextents 2 optimal 1m);**

erstes Extent : 250K; zweites ebenfalls (es gibt zwei Extents)

DBA – Rollback-Segmente

- *Die View DBA_ROLLBACK_SEGS*

- SQL>**desc dba_rollback_segs**
zeigt den Aufbau der View an

```
SQL>SELECT TABLESPACE_NAME, SEGMENT_NAME, STATUS  
FROM DBA_ROLLBACK_SEGS;
```

zeigt u.a. den Status (`online`, `offline`).

DBA – Rollback-Segmente

- *alter rollback segment online*

- **SQL>ALTER ROLLBACK SEGMENT R77 ONLINE;**

bringt das Rollback-Segment R77 wieder online.

```
SQL>SELECT TABLESPACE_NAME, SEGMENT_NAME, STATUS  
FROM DBA_ROLLBACK_SEGS;
```

zeigt diesen Status (online, offline).

- **Rollback-Segmente manuell verkleinern oder löschen**

- **SQL>ALTER ROLLBACK SEGMENT rbs1 SHRINK TO 100K;**

```
SQL>DROP ROLLBACK SEGMENT R77;
```

DBA – Redo Logs

- **Das Multiplexen von Redo Logs**

- **SQL>CREATE DATABASE „test“**

- LOGFILE GROUP 1 (,/d0/oradata/redo0101.log‘,**

- /d0/oradata/redo0102.log‘)**

- SIZE 500K REUSE,**

- GROUP 2 (,/d0/oradata/redo0201.log‘,**

- ,/d0/oradata/redo0202.log‘)**

- SIZE 500K REUSE**

- DATAFILE ,/d0/oradata/system_001.dbf‘**

- SIZE 250M REUSE**

- MAXLOGFILES 20**

- CONTROLFILE REUSE**

- MAXDATAFILES 500;**

- Redo0101.log und redo0102.log enthalten die selben Transaktionen (...02... Enthält eine Kopie der Transaktion in ..01..).

DBA – Redo Logs

- **Löschen eines Redo Logs**

- **1 Schritt** : prüfen ob die Gruppe nicht **CURRENT** ist
 - **SQL>SELECT group#, thread#, bytes, members, status FROM V\$LOG;**
- **2 Schritt** : Logwechsel erzwingen
 - **SQL>ALTER SYSTEM SWITCH LOGFILE;**
- **evtl. nochmals prüfen (Schritt 1)**
- **3. Schritt** : löschen des Redo Logs
 - **SQL>ALTER DATABASE DROP LOGFILE GROUP 1;**
- **evtl. nochmals prüfen (Schritt 1)**

DBA – Redo Logs

- **Ein Redo Log hinzufügen**

- **SQL>ALTER DATABASE ADD LOGFILE GROUP 4
(,/d0/oradata/redo04.log',
 ,/d0/oradata/redo04a.log')
SIZE 1M REUSE;**

DBA – Steuerdateien

- **Enthält**
 - **Den Datenbanknamen**
 - **Namen und Standardort der Datendateien**
 - **Namen und Standardort der Redo Logs**
 - **Datumsangaben und Zeitstempel für alle Datendateien**
 - **Datumsangaben und Zeitstempel für alle Redo Logs**
- **Steuerdatei anlegen**
 - **Diese wird beim CREATE DATABASE Befehl angelegt**
 - **SQL>SHOW PARAMETERS CONTROL_FILES**
beschreibt den Aufbau der Control-Datei
(siehe auch init.ora)

DBA – Benutzerkonto einrichten

- **SQL>CREATE USER mich IDENTIFIED BY meinemPasswort DEFAULT TABLESPACE users QUOTA 10m ON users TEMPORARY TABLESPACE temp QUOTA UNLIMITED ON users;**
- Der Benutzer erhält 10 Megabytes Speicher im Tablespace users
- Zusätzlich darf er für SORT, ... temporär Speicher verwenden (unlimitiert).
- **SQL>GRANT CONNECT, RESOURCE TO mich;**
 - Jetzt darf ich sogar einloggen!
- **SQL>SELECT * FROM DBA_USERS;** // zeigt alle Angaben



Oracle für Einsteiger

Aufbaukurs

Aufbaukurs

- Mehr SQL*Plus
- Mehr PL/SQL
- Mehr zur Datenbankverwaltung
- Oracle Enterprise Manager
- Verteilte Verarbeitung

Aufbaukurs

Mehr SQL*Plus

Terminologie

- include-Datei
 - SQL*Plus Skripts können weitere Skripts aufrufen, “includen/einbeziehen”
- SQL*Plus Skripts
 - bestehen aus SQL Anweisungen plus Steueranweisungen z.B. für die Formattierung.

SQL*Plus – Einsatz in der Produktionsumgebung

- Kommentare
 - Zeilenweise
 - Rem das ist eine Kommentarzeile
 - -- der Rest ist Kommentar
 - /*
das ist alles Kommentar
bis hier */

SQL*Plus – Einsatz in der Produktionsumgebung

- include-Dateien

- Beispiel

- meinSkript.sql

- Rem -----

- Rem Formattierungen hole ich mir anderswo

- Rem -----

- @formattierung.sql – da stehen alle Formattierungen drin

- Rem (include Datei formattierung.sql)

- spool meineAusgabe.txt

- Rem überschreiben einer Formattierung

- column Name heading 'Student' format a25

- select * from students;

- spool off – Ausgabe in Datei ist beendet

- exit

SQL*Plus – Einsatz in der Produktionsumgebung

- Union (Vereinigung), Intersect (Durchschnitt), Minus (Differenz)

- SQL> select * from table_1
 union
 select * from table_2;

selektiert alle Elemente aus table_1 und table_2, wobei mengentheoretisch korrekt keine Zeile doppelt erscheint.

- Union All

- SQL> select * from Tabelle_1
 union all
 select * from Tabelle_2;

selektiert alle Elemente und zeigt auch mehrfache Elemente mehrfach an.

SQL*Plus – Einsatz in der Produktionsumgebung

- Union (Vereinigung), Intersect (Durchschnitt), Minus (Differenz)

- SQL> select * from table_1
minus
select * from table_2;

selektiert alle Elemente aus table_1, die nicht in table_2 sind.

- Intersect

- SQL> select * from Tabelle_1
intersect
select * from Tabelle_2;

selektiert alle Elemente, die sowohl in Tabelle_1 als auch Tabelle_2 sind.

- **ACHTUNG:** Mengenoperationen verlangen gleiche Datentypen
 - Man kann nicht Äpfel mit Autos mischen

SQL*Plus – Befehlszeileneditor

- Einfache Editorbefehle

- A append
- C/alt/neu/ von alt in neu ändern
- DEL löschen der aktuellen Zeile
- L <nr> Zeile <nr> anzeigen (UND setzen)

- Einsatz anderer Editoren

- **define_editor** = 'meinEditor'

aktiviert meinen Wahnsinns-Editor, den nur ich kenne (vi_me).

SQL*Plus – Die Tabelle *dual*

- dual
 - Ist eine einzeilige Tabelle, die immer zur Verfügung steht
 - Anwendungsbeispiel
 - SQL>**select 12*34*33398 + 12343 from dual;**

liefert das Ergebnis der math. Formel

SQL*Plus – Oracle-Funktionen

- Mathematische Funktionen
 - Es gibt so ziemlich alles, was im DB Umfeld Sinn macht
 - +, -, *, /
 - Abs()
 - Cos()
 - ...
- String-Funktionen
 - Auch hier hat man eine riesige Auswahl
 - Length(...)
 - Lower(...)
 - ...
- Datums-Funktionen
 - Lässt nichts zu wünschen übrig
 - Add_month(,,,), months_between(d1,d2) –liefert die Anzahl Monate zwischen d1, d2
 - ...

SQL*Plus – Oracle-Funktionen

- **group by** Funktionen
 - Gruppenbildung (z.B. : alle Studenten einer Klasse) ermöglicht spezielle Funktionen
 - **Avg()** : Durchschnitt eines Wertfeldes der Gruppe
 - **Count** : Anzahl Zeilen der Gruppe (z.B. Anzahl Studenten pro Klasse)
 - **Max()**,
 - **Min()**,
 - **Stddev()** (Standardabweichung),
 - **variance()**

SQL*Plus – Oracle-Funktionen

- Generieren von SQL mit SQL
 - SQL>Rem zuerst wird die Spool-Datei angegeben (Ausgabedatei)
 - SQL>spool countall.sql
 - SQL>Rem jetzt generieren wir SQL
 - SQL>select ,select count(*) from ' || table_name ||';
 from user_tables;
 - SQL>Rem jetzt führen wir die Ergebnisdatei aus
 - SQL>@countall.sql

SQL*Plus – Oracle-Funktionen

- decode-Anweisung
 - Erlaubt eine **if...then...else** Abfrage in SQL*PLUS
 - Syntax
 - **Decode (column, comparison, action, comparison,action...,else action)**
 - Beispiel
 - **Update table_1**
set category =
decode(category,
,A', ,a',
,B', ,b'
,*');