|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Grundlagen**  **DB**: Datenbasis, strukturierte Daten auf HDD  **DBMS**: Datenbankmanagement System  ***Redundanzfreiheit****: Element nur 1x pro DB,* ***Datenintegrität****: Datenkonsistenz*: korrekte Daten *Datensicherheit*: vor phy. Verlust*, Datenschutz*: vor unberecht. Zugriff  **DBS** = DBMS + (n \*) DB  **Funktionen DBMS** Transaktionen, Mehrbenutzerbetrieb, Sicherheit, Backup & Recovery, Generische Datenstrukturen  **Vorteile DBMS** Skalierbar, Sicherheit, Integrität, Live-Abfragen, Kapslung  **Datenbankmodelle** Hierarchisch., relational, Netzwerk, Objektrelational, -orientiert, NOSQL  **Datenbankentwurf**   1. Anforderungsanalyse -> Anforderunsspezifikation 2. Konzeption. Entwurf -> konzeptuelles Schema(UML)  * Unabhängig von eingesetztem DBMS  1. Logischer Entwurf -> logisches Schema  * Rel. Schreibweise, angepasst an DBMS  1. Physischer Entwurf -> Physisches Schema  * SQL-Code (+ Hardwareplanung)   **1 Tier:** DBMS gleicher Prozess wie Client  Bildschirmausschnitt**2 Tier** gleiche Maschine  **UML: Objekt-, Klassendiagramm**  Bildergebnis für uml generalisierung icoon  **Disjoint:** 1 Obj. = 1 Subcl. **Overlapping:** 1 Obj = n Sub. **Complete:** All Subclas. def. **Incomplete:** More sub. poss. **Abbildungsregel/strategie**  ***1 Tabelle pro Sub- und Superklasse***   * Redundanzfreiheit(«reine Lösung»)[auch bei overlapping] * Komplexe Zugriffe(«wegen vielen Tabellen»)   ***2. Je eine Tabelle pro Subklasse. PK = FK ref. Superclass***   * Einfache Zugriffe auf die Elemente der Subklasse * Komplexe Abfragen über alle Elemente der Superklasse   ***3. Eine Superklasse mit allen möglichen Attributen(NULL)***   * Einfache Zugriffe & effiziente Zugriffe * 3. NF verletzt(🡪Abhängigkeiten innerhalb Tabelle)   **ANSI 3:** externe, konzeptionelle, interne  **Schlüsselkandidaten:** Attribut oder Attributkombination, welche Datensatz eindeutig identifizierbar macht  Eindeutig, laufend zuteilbar, knapp, invariant  **Surrogatschlüssel:** künstlicher Schlüssel  **Referentielle Integrität:** FK NN🡪Datenkonsis.  **Einfügeanomalie, Löschanomalie, Änderungsanomalie**  **1. NF:** Attribute sind atomar(z.B nicht Adresse)  **2. NF:** Jedes Nichtschlüsselattribut voll funktional abhängig von Schlüsselkandidat  **3. NF:** Jedes NSA transitiv abhängig(A🡨C, A🡨B🡨C)  Boyce-Codd-NF, 4. & 5. NF  **Relationales Modell / Rel. Schreibweise**  practice (practiceID INT PK, name TEXT UNIQUE, day DATE NOT NULL, *teamid* INT FK REFERENCES team on delete restrict);  ***Windows Function*** **MIN(), MAX(), AVG(), SUM() OVER: Mehr verdienen als Durchschnittsabteilungslohn** **SELECT** name, salaer > avg(salaer) OVER(PARTITION BY abtnr) as "Mehr als Durchschnittslohn" **FROM** angestellter; **Rank()OVER: Option ORDER zwingend!** Bestverdiener pro Abt **SELECT** name, salaer, sub **FROM** (**SELECT** name, salaer, **RANK()** **OVER** (**PARTITION** **BY** AbtNr order by salaer desc) as sub from angestellter) as super where sub < 4 ;  ***Common Table Expression(CTE) – WITH-Queries***  **Ohne Rekursion:** Ausgabe von Zeiterfassungen  **WITH** zeiterfassung **AS** **(**  SELECT name, bezeichnung, zeitanteil FROM angestellter a inner join projektzuteilung pz on a.persnr = pz.persnr inner join projekt p on pz.projnr = p.projnr  **)**  SELECT name AS "Mitarb.", bezeichnung AS "Projekt", zeitanteil AS "Zeit" FROM zeiterfassung order by 1,2;  **Mit Rekursion**: Ausgabe von Untergebene(Hierarchie**)**  **WITH** **RECURSIVE** untergebene(persnr, name, chef, stufe) **AS**  (Select persnr, name, chef, 1 from angestellter where persnr = 1010  **UNION ALL**  select a.persnr, a.name, b.persnr, b.stufe + 1 from angestellter a INNER JOIN untergebene b on a.chef = b.persnr where b.stufe < 3)  select \* from untergebene;  ***Views***  **Updatable**: wenn max 1 Table+kein Select Distinct.(Sonsts Trigger, Stored Proc.)  **CREATE** **VIEW** SalaerProAbteilung **AS** SELECT ab.name, sum(salaer)from angestellter a inner join abteilung ab on a.abtnr = ab.abtnr group by ab.abtnr;  Select \* from SalaerProAbteilung;  **DROP** **VIEW** SalaerProAbteilung;  **Data Control Language(DCL):** *ROLE, GRANT, REVOKE*  **CREATE** **ROLE** *username* **WITH** **LOGIN** **PASSWORD** ‘*password’*;  **CREATE** **ROLE** username WITH LOGIN PASSWORD ‘password’ **WITH** C**REATEDB**;  **ALTER** **ROLE** username **WITH** **CREATEROLE**;  **GRANT** SELECT, INSERT **ON** **TABLE** *testtable* **TO** testuser [WITH GRANT OPTION];  **GRANT** **SELECT** **ON** *testdb* to *testuser*;  **DROP ROLE** *username***;**  **ALTER ROLE** *username* **WITH PASSWORD** ‘*password*’;  **GRANT** *groupname* **TO** *username*;  **Berechtigungen:**SELECT, INSERT, UPDATE, DELETE, ALL  **Systemprivilegien:** CREATEROLE(DB + ROLE), CREATEDB  **Transaktionen** **/ Transaction Control Language(TCL)**  **Atomicity**: Operationen werden vollständig oder gar nicht ausgeführt  **Consistency**: Daten sind vor wie nachher konsistent.  **Isolation**: Transaktionen isoliert voneinander  **Durability**: Änderungen sind nicht flüchtig. | **Structured Query Language(SQL)**  **Data Defintion Language(DDL):** *CREATE ALTER DROP*  **ON DELETE**   * **CASCADE:** löscht Elemente, die darauf referenzieren * **RESTRICT(DEFAULT):** Bei Referenzen, Fehler/Rollback * **SET NULL** * **SET DEFAULT**   **CREATE DB** somedb owner someuser;  **CREATE** **TABLE** angestellter(persnr INTEGER PRIMARY KEY, name VARCHAR(80) NOT NULL UNIQUE, abteilung INTEGER REFERENCES abteilung);  **PRIMARY KEY**(att1, att2);  **ALTER** **TABLE** angestellter **ADD** **CONSTRAINT** abteilung **FOREIGN** **KEY** (abteilung) REFERENCES abteilung [ON DELETE CASCADE];  **ALTER** **TABLE** angestellter **ADD** **CONSTRAINT** check\_PLZ **CHECK**(plz BETWEEN 1000 AND 9999);  **DROP CONSTRAINT** check\_PLZ;  **ALTER TABLE** ang **ADD** **COLUMN** street varchar(30);  **ALTER TABLE** ang **DROP** **COLUMN** street [RESTRICT];  **ALTER TABLE** ang **ALTER** **COLUMN** street TYPE varchar(80);  **ALTER TABLE** ang **RENAME** **COLUMN** street to streetname;  **DROP TABLE** [IF EXISTS] ang [CASCADE | RESTRICT] **Data Manipulation Language(DML)** *INSERT INTO, UPDATE SET, DELETE /SELECT FROM* **Bedingungen:**   |  |  | | --- | --- | | * BETWEEN 10 AND 20 | * IN(‘Zug ‘Chur’,) | | * NOT | * IS NULL | | * (I)LIKE | * IS NOT NULL |   **%** 🡪 beliebige und beliebig viele Character **\_** 🡪 genau ein beliebiger Character  **SELECT** pid as Personalnummer **FROM** angestellter **WHERE** name LIKE '%a' OR name ILIKE 'e%' order by 1 desc limit 2;  **SELECT** **DISTINCT** wohnort **FROM** angestellter;  **SELECT** sum(salaer) as Abteilungslohn **FROM** angestellter **GROUP BY** abtnrORDER BY 1 desc **;**  **SELECT** abtnr**,** avg(salaer**)**asDurchschnittslohn **FROM** angestelltergroupbyabtnr **HAVING** avg(salaer) >= 5000 order by avg(salaer) desc**;**   * **Aggregat Funktionen**   SUM(), AVG(), MIN(), MAX()  **Andere wichtige Funktionen**  COUNT(), ROUND(), MOD()[Modulo], NOW()[aktuelles Datum],  ***JOINS***  **SELECT** a.name as Angestellter, ab.name as Abteilung **FROM** angestellter **a** **INNER** **JOIN** abteilung ab on **a**.abtnr = **ab**. abtnr;  Bildschirmausschnitt  ***Befehle***  SET TRANSACTION ISOLATION LEVEL <LEVEL>  SHOW TRANSACTION ISOLATION LEVEL;  BEGIN [TRANSACTION ISOLATION LEVEL <LEVEL>];  [SAVEPOINT SavepointName1]  ROLLBACK [SavepointName1] COMMIT;  ***Serialisierbarkeit***    BildschirmausschnittBildschirmausschnitt***Konfliktpaare(sobald ein write vorkommt)***    ***Isolation Level***  Bildschirmausschnitt  **1.**Lesezugriffe nicht synchronisiert(keine Read Locks) **2.**Lesezugriffe nur kurz / temporär synchronisiert(DEFAULT) **3.**Einzlene zugriffe Rows sind synchronisiert **4.**Nacheinander(Sequentialisiert) Änderungen erst wenn andere Transaktion abgeschlossen || **a**. Lese Daten von anderen nicht committed Transaktion **b.** Lese gleiche Daten mehrmals, sehe aber andere Werte **c.** Select in table, entdecke plötzlich neue gelöschte Rows. Insert neue Snapshot IS: T sieht einen Snapshot zu Startzeitpunkt. Vergleich Commit.  ***Locking***  **Wichtig:** Garantiert keine Serialisierbarkeit  **2PL**: Growing Phase(lock), Shrinking Phase(unlock)  Garantiert Serialisierbarkeit, DL & CR möglich Bildschirmausschnitt**Deadlocks** Transaktionen, haben sich ausgesperrt  **Betriebsmittel/Wartegraph**  **BildschirmausschnittDeadlock Behebung**  BildschirmausschnittOpfer bestimmen oder Timeout definieren  **Cascading Rollbacks**  **S2PL:** Alle unlocks erst nach Ende Transaktion. Dead Locks möglich, keine Cascading Rollbacks  WAL(Write Ahead Log): Änderungen der Transaktionen, Checkpoints | Bildergebnis für self joinBildergebnis für cross joinBildschirmausschnittBildschirmausschnittBildschirmausschnitt  **SELECT** a.name, a.salaer, b.name, b.salaer **FROM** angestellter a **INNER** **JOIN** angestellter b **on** a.chef = b.persnr where a.salaer > b.salaer;  **SELECT** a.name, ab.name **FROM** angestellter a **CROSS JOIN** abteilung ab order by a.name;  **SELECT** a.name, ab.name **FROM** angestellter a FULL [OUTER] JOIN abteilung ab on a.abtnr = ab.abtnr;  **SELECT** zeitanteil, name **FROM** projektzuteilung pz **RIGHT** [OUTER] **JOIN** angestellter a on pz.persnr = a.persnr;  **SELECT** p.bezeichnung, a.name **FROM** projekt p **LEFT** [OUTER] **JOIN** angestellter a on a.persnr = p.ProjLeiter;  Full (Outer) Join  Right (Inner) Join  Right Outer Join  Left Outer Join  Left (Inner) Join  BildschirmausschnittBildschirmausschnitt  **Natural Join**: Join wenn gleichnamige Attribute **Theta** **Join**: Relationale Algebra(<,>,=,≥,≤) **Equi** **Join**: Join wenn etwas gleich(«equals-Join) Lateral Join: Auf äusser/innere Attr. zugreifen  Unterabfragen  Korreliert: Unterabfrage greif auf Attribute der äusseren Abfrage zu 🡪 funktioniert nur mit äusseren Abfrage  Unkorreliert: Unterabfrage greift nicht auf Attribute der äusseren Abfrage 🡪 funktioniert ohne äussere Abfrage  Lateral Join(zugreifen auf äussere/innere Elemente)  **SELECT** ab.name,eintritt.name, eintritt.eintrittsdatum **FROM** abteilung ab **cross** **join** **lateral**( Select \* from angestellter a where a.abtnr = ab.abtnr order by eintrittsdatum asc limit 2) eintritt;  IN : Alle Abteilungsleiter ausgeben  **SELECT** name **FROM** angestellter where persnr **IN**( select AbtChef from abtleitung);  EXISTS: Alle die an einem Projekt arbeiten(keine Duplikate)  **SELECT** name **FROM** angestellter a where **EXISTS**( select \* FROM projektzuteilung pz where a.persnr = pz.persnr);  ANY: Angestelle, welche mehr verdienen als irgendein Chef  **SELECT** name **FROM** angestellter a where salaer > **ANY(** select salaer from angestellter where chef is null);  ALL: MA Abt 3, welche Lohn > Lohn von allen MA von Abt 3  **SELECT** name **FROM** angestellter where abtnr = 2 and salaer > **ALL**(select salaer from angestellter where abtnr = 3);  Mengenoperationen(Gleiche Anzahl Spalten!)  UNION: Verbindet zwei Mengen-> Essensgeld der Abteilungen  **SELECT** a.name, '9' as Essensgeld **FROM** angestellter a where abtnr = 1 **UNION** [ALL] select a.name, '0' as Essensgeld from angestellter a where abtnr = 2;  EXCEPT: Linke Menge ohne rechte Menge -> MA ohne Projekt  **SELECT** persnr, name **FROM** angestellter **EXCEPT SELECT** persnr, projnr::varchar **FROM** projektzuteilung;  INTERSECT: Schnittmenge 🡪 Abteilungen welche nicht leer  **SELECT** abtnr **FROM** abteilung **INTERSECT SELECT** abtnr **FROM** angestellter;  Read Committed: Zeile 6 = (10,21) / Zeile 8 = (20,21) / Zeile 10 = (20, 21)  Repeatable Read:  Zeile 6 = (10,21) / Zeile 8 = (10,21) / Zeile 10 = (20, 21)  🡪T1 sieht Änderungen von T2 erst wenn T1 Committ hat.  Index / Performance Optimierung  Index Algorithmen: ISAM, B-Tree, Hash., Bitmap, Mehrdimensionale Indexe  Postgres: B-Tree, Hash, BRIN, GiST, Gin  Varianten: Clustered, Primär/Sekundär, Zusammengesetzter, Partieller, Funktionaler  **CREATE INDEX <IndexName> ON <Table(attr)> [USING indexmethod(col)];**  **DROP INDEX <IndexName>;**  log .Optimierung: Heuristiken🡪 Schema ok, Optimiere Abfrage, kleines Zwischenresult  Phy. Optimierung: Optimaler Index, Konf an HW angepasst, Systemarchitektur IOPS,…  Bildschirmausschnitt**JDBC**  BildschirmausschnittBildschirmausschnittBildschirmausschnitt  Bildschirmausschnitt |