# Select aller Tabellen

|  |  |
| --- | --- |
| **SELECT** \* **FROM** Produkt;  C:\Users\Lars\Dropbox\Studium\Semester 2\Datenbanken\Praktikum\Praktikum 4\img\select_produkt.png | |
| **SELECT** \* **FROM** Kunde;  C:\Users\Lars\Dropbox\Studium\Semester 2\Datenbanken\Praktikum\Praktikum 4\img\select_kunde.png  **SELECT** \* **FROM** Kondition;  C:\Users\Lars\Dropbox\Studium\Semester 2\Datenbanken\Praktikum\Praktikum 4\img\select_kondition.png | **SELECT** \* **FROM** Bestellung;  C:\Users\Lars\Dropbox\Studium\Semester 2\Datenbanken\Praktikum\Praktikum 4\img\select_bestellung.png  **SELECT** \* **FROM** Bestellposition;  C:\Users\Lars\Dropbox\Studium\Semester 2\Datenbanken\Praktikum\Praktikum 4\img\select_bestellposition.png |
| **SELECT** \* **FROM** Lieferant;  C:\Users\Lars\Dropbox\Studium\Semester 2\Datenbanken\Praktikum\Praktikum 4\img\select_lieferant.png | |

# Trigger anlegen

|  |  |
| --- | --- |
| -- Bestellung gesamtpreis hinzufügen  **ALTER TABLE** Bestellung **ADD** (  gesamtpreis **DECIMAL**(10,2)  **DEFAULT** 0 **NOT NULL**  );  -- gesamtpreis für bereits bestehende Bestellpositionen berechnen  **UPDATE** Bestellung b  **SET** gesamtpreis = (  **SELECT SUM**(preis\*menge)  **FROM** Bestellposition bp  **WHERE** b.bID=bp.bID  ); | -- Die Tabelle Bestellung danach  **SELECT** \* **FROM** Bestellung;  C:\Users\Lars\Dropbox\Studium\Semester 2\Datenbanken\Praktikum\Praktikum 7\img\select_1_bestellung_nach_migration.png |

-- Anlage des Triggers

**CREATE TRIGGER** bestellung\_gesamtpreis

**AFTER INSERT OR UPDATE OR DELETE ON** Bestellposition

**FOR EACH ROW**

**BEGIN**

-- ON INSERT

**IF** :OLD.bID **IS NULL THEN**

**UPDATE** Bestellung b

**SET** gesamtpreis = gesamtpreis + :NEW.preis \* :NEW.menge

**WHERE** b.bID = :NEW.bID;

**END IF**;

-- ON DELETE

**IF** :NEW.bID **IS NULL THEN**

**UPDATE** Bestellung b

**SET** gesamtpreis = gesamtpreis - :OLD.preis \* :OLD.menge

**WHERE** b.bID = :OLD.bID;

**END IF**;

-- ON UPDATE

**IF** :OLD.bID **IS NOT NULL AND** :NEW.bID **IS NOT NULL THEN**

**UPDATE** Bestellung b

**SET** gesamtpreis = gesamtpreis - (:OLD.preis \* :OLD.menge) + (:NEW.preis \* :NEW.menge);

**WHERE** b.bID = :NEW.bID;

**END IF**;

**END**;

# Explain Plan (SQLDeveloper)

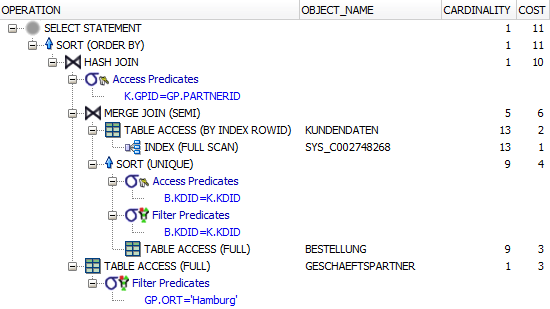
**SELECT** \*

**FROM** Kunde k

**WHERE** ort='Hamburg'

**AND EXISTS** (**SELECT** \* **FROM** Bestellung b **WHERE** k.kdID=b.kdID)

**ORDER BY** k.vorname, k.nachname;



Der Explain-Plan gibt die interne Umsetzung der an die Datenbank gestellten Befehle dar.

In unserem Beispiel sieht man das für eine voll/qualifizierte Abfrage auf unsere View Kunde zuerst ein Join zwischen den Tabellen Kundendaten und Geschaeftspartner geschehen muss (HASH JOIN), da einige Kundendaten in der Geschaeftspartnertabelle ausgelagert sind. Auch zwischen den Tabellen Kunde und Bestellung geschieht ein Join (MERGE JOIN (SEMI)).

Dabei wird auch gezeigt welche Einschränkungen bei dem Zugriff gelten, wie z.B. Zugriffs (JOIN-Bedingung)- und Filterprädikate (WHERE-Bedingung).