Solution

Solution

Solution 5a

```
CREATE VIEW PromotionSummary AS
        SELECT category, MIN(price) AS minprice, MAX(price) AS maxprice FROM
           Books
        WHERE promoted
        GROUP BY category;
```

Solution 5b

```
CREATE OR REPLACE FUNCTION demoteBooks() RETURNS TRIGGER AS $$
BEGIN
        UPDATE Books SET promoted = False WHERE category = OLD.category;
END
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER demoteBooksTrigger INSTEAD OF DELETE ON PromotionSummary
    FOR EACH ROW
    EXECUTE PROCEDURE demoteBooks();
```

it is acceptable to shorten this to:

```
demoteBooks() → UPDATE Books SET promoted = False WHERE category = OLD.
   category;

CREATE TRIGGER demoteBooksTrigger INSTEAD OF DELETE ON PromotionSummary
    FOR EACH ROW
    EXECUTE PROCEDURE demoteBooks();
```

Alternative Solution

If we do not interpret that the minimal and maximal price should be computed per category:

Solution 5a

```
CREATE VIEW PromotionSummary AS (
        SELECT DISTINCT category,
                (SELECT MIN(price) FROM Books WHERE promoted) AS minprice,
                (SELECT MAX(price) FROM Books WHERE promoted) AS maxprice
        FROM Books
);
```

Solution 5b

```
CREATE OR REPLACE FUNCTION DemoteCategory() RETURNS TRIGGER AS $$
        BEGIN
                UPDATE Books SET promoted = FALSE WHERE category = OLD.
                   category;
                RETURN OLD;
        END;
$$ LANGUAGE plpgsql;

DROP TRIGGER IF EXISTS UpdatePromotion ON PromotionSummary;
CREATE TRIGGER UpdatePromotion INSTEAD OF DELETE ON PromotionSummary
        FOR EACH ROW
        EXECUTE PROCEDURE DemoteCategory();
```