Bazy danych - sprawozdanie z projektu

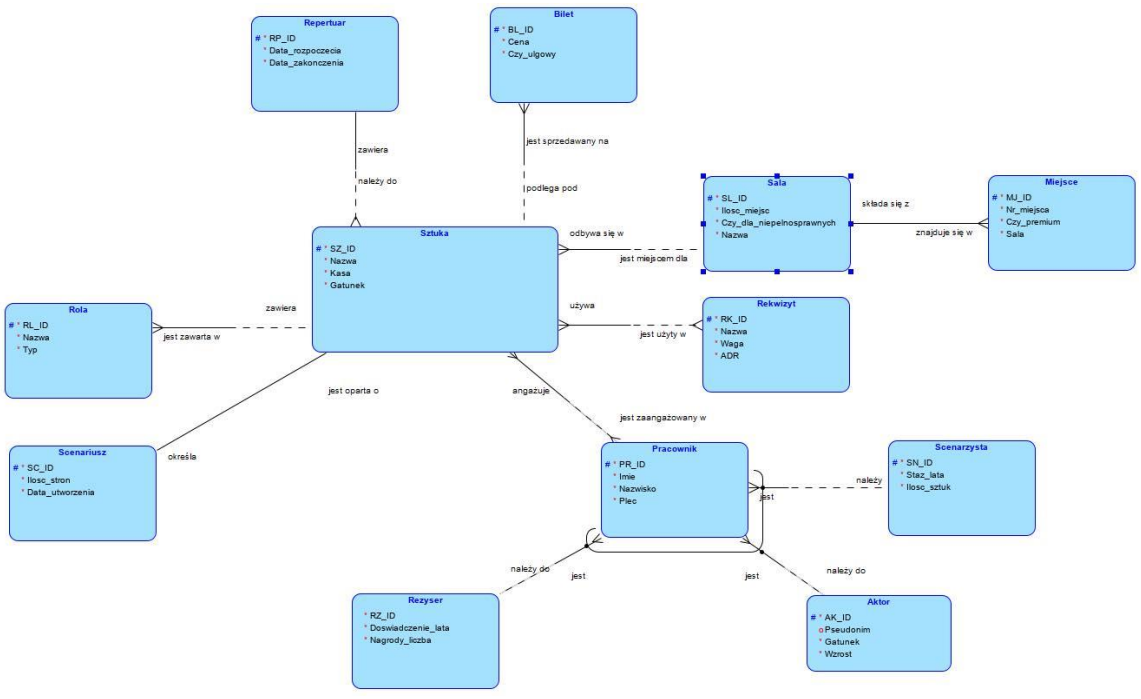
1. Podstawowe informacje

- **Autorzy projektu:** Natalia Janik, Agata Konarska, Michał Skowronek, Konrad Wasiak.....
- **Nazwa grupy:** Informatyka w Gospodarce i Administracji .....
- **Temat:** Teatr .....
- **Wybrany zakres (ocena): bardzo dobry** .....
- **Baza danych:** ie83758.....

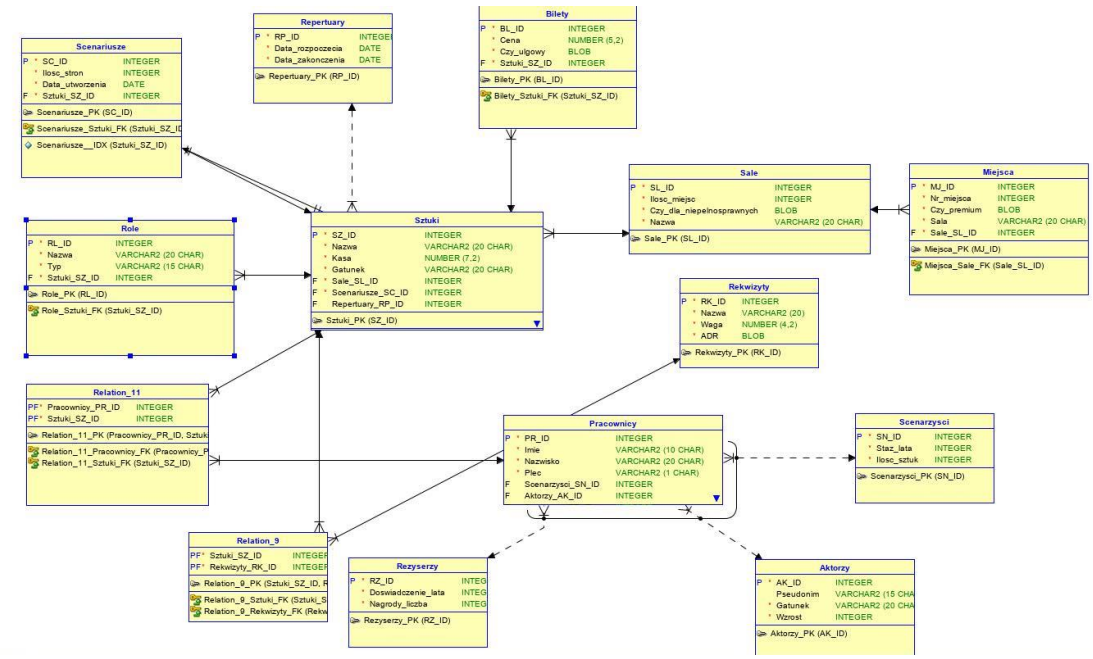
2. Wprowadzenie – opis studium przypadku

Opracowano bazę danych dla teatru. Teatr zatrudnia aktorów, reżyserów i scenarzystów. Prowadzony jest rejestr pracowników. W ramach repertuaru teatru, który zmienia się cyklicznie odgrywane są sztuki. Każda sztuka powstaje w oparciu o scenariusz; występują w niej role, w które wcielają się aktorzy. Oprócz aktorów w sztukach występują rekwizyty. Teatr prowadzi ewidencję rekwizytów, gromadząc informacje o tym, czy mogą być niebezpieczne i stanowić zagrożenie. W teatrze znajdują się sale mieszczące różną ilość osób. Niestety część z nich nie jest przystosowana dla osób z niepełnosprawnościami. Niektóre miejsca w salach teatru zapewniają lepszy widok na scenę, są oznaczone jako miejsca „premium”. Na sztukę obowiązuje bilet, na który można dostać rabat w postaci ulgi np. na podstawie okazania legitymacji szkolnej. Teatr ewidencjonuje sprzedane bilety.

3. Model związków encji



4. Model relacyjnej bazy danych



5. Schemat bazy – tabele

```
CREATE TABLE aktorzy (
    ak_id       INTEGER NOT NULL,
    pseudonim   VARCHAR2(15 CHAR),
    gatunek     VARCHAR2(20 CHAR) NOT NULL,
    wzrost      INTEGER NOT NULL
);

ALTER TABLE aktorzy ADD CONSTRAINT aktorzy_pk PRIMARY KEY ( ak_id );

CREATE TABLE bilety (
    bl_id          INTEGER NOT NULL,
    cena           NUMBER(5, 2) NOT NULL,
    czy_ulgowy     BLOB NOT NULL,
    sztuki_sz_id   INTEGER NOT NULL
);

ALTER TABLE bilety ADD CONSTRAINT bilety_pk PRIMARY KEY ( bl_id );

CREATE TABLE miejsca (
    mj_id         INTEGER NOT NULL,
    nr_miejsca    INTEGER NOT NULL,
    czy_premium   BLOB NOT NULL,
    sala          VARCHAR2(20 CHAR) NOT NULL,
    sale_sl_id    INTEGER NOT NULL
);

ALTER TABLE miejsca ADD CONSTRAINT miejsca_pk PRIMARY KEY ( mj_id );

CREATE TABLE pracownicy (
    pr_id                    INTEGER NOT NULL,
    imie                     VARCHAR2(10 CHAR) NOT NULL,
```

```
    nazwisko                VARCHAR2(20 CHAR) NOT NULL,
    plec                    VARCHAR2(1 CHAR) NOT NULL,
    scenarzysci_sn_id       INTEGER,
    aktorzy_ak_id           INTEGER,
    rezyserzy_rz_id         INTEGER,
    ak_id                   INTEGER NOT NULL,
    rezyserzy_rezyserzy_id  NUMBER
);

ALTER TABLE pracownicy
    ADD CONSTRAINT arc_1 CHECK ( ( ( rezyserzy_rz_id IS NOT NULL )
                                   AND ( rezyserzy_rezyserzy_id IS NOT NULL )
                                   AND ( aktorzy_ak_id IS NULL )
                                   AND ( scenarzysci_sn_id IS NULL ) )
                                 OR ( ( aktorzy_ak_id IS NOT NULL )
                                      AND ( rezyserzy_rz_id IS NULL )
                                      AND ( rezyserzy_rezyserzy_id IS NULL )
                                      AND ( scenarzysci_sn_id IS NULL ) )
                                 OR ( ( scenarzysci_sn_id IS NOT NULL )
                                      AND ( rezyserzy_rz_id IS NULL )
                                      AND ( rezyserzy_rezyserzy_id IS NULL )
                                      AND ( aktorzy_ak_id IS NULL ) ) );

ALTER TABLE pracownicy ADD CONSTRAINT pracownicy_pk PRIMARY KEY ( pr_id );

CREATE TABLE rekwizyty (
    rk_id  INTEGER NOT NULL,
    nazwa  VARCHAR2(20) NOT NULL,
    waga   NUMBER(4, 2) NOT NULL,
    adr    BLOB NOT NULL
);

ALTER TABLE rekwizyty ADD CONSTRAINT rekwizyty_pk PRIMARY KEY ( rk_id );

CREATE TABLE relation_11 (
    pracownicy_pr_id  INTEGER NOT NULL,
    sztuki_sz_id      INTEGER NOT NULL
);

ALTER   TABLE   relation_11   ADD   CONSTRAINT   relation_11_pk   PRIMARY   KEY   (
pracownicy_pr_id,
                                                                    sztuki_sz_id
);

CREATE TABLE relation_9 (
    sztuki_sz_id     INTEGER NOT NULL,
    rekwizyty_rk_id  INTEGER NOT NULL
);

ALTER TABLE relation_9 ADD CONSTRAINT relation_9_pk PRIMARY KEY ( sztuki_sz_id,
                                                                 rekwizyty_rk_id
);

CREATE TABLE repertuary (
    rp_id             INTEGER NOT NULL,
    data_rozpoczecia  DATE NOT NULL,
    data_zakonczenia  DATE NOT NULL
);
```

```
ALTER TABLE repertuary ADD CONSTRAINT repertuary_pk PRIMARY KEY ( rp_id );

CREATE TABLE rezyserzy (
    rezyserzy_id        NUMBER NOT NULL,
    rz_id               INTEGER NOT NULL,
    doswiadczenie_lata  INTEGER NOT NULL,
    nagrody_liczba      INTEGER NOT NULL
);

ALTER TABLE rezyserzy ADD CONSTRAINT rezyserzy_pk PRIMARY KEY ( rz_id,
                                                                rezyserzy_id );

CREATE TABLE role (
    rl_id         INTEGER NOT NULL,
    nazwa         VARCHAR2(20 CHAR) NOT NULL,
    typ           VARCHAR2(15 CHAR) NOT NULL,
    sztuki_sz_id  INTEGER NOT NULL
);

ALTER TABLE role ADD CONSTRAINT role_pk PRIMARY KEY ( rl_id );

CREATE TABLE sale (
    sl_id                       INTEGER NOT NULL,
    ilosc_miejsc                INTEGER NOT NULL,
    czy_dla_niepelnosprawnych   BLOB NOT NULL,
    nazwa                       VARCHAR2(20 CHAR) NOT NULL
);

ALTER TABLE sale ADD CONSTRAINT sale_pk PRIMARY KEY ( sl_id );

CREATE TABLE scenariusze (
    sc_id            INTEGER NOT NULL,
    ilosc_stron      INTEGER NOT NULL,
    data_utworzenia  DATE NOT NULL,
    sztuki_sz_id     INTEGER NOT NULL
);

CREATE UNIQUE INDEX scenariusze__idx ON
    scenariusze (
        sztuki_sz_id
    ASC );

ALTER TABLE scenariusze ADD CONSTRAINT scenariusze_pk PRIMARY KEY ( sc_id );

CREATE TABLE scenarzysci (
    sn_id        INTEGER NOT NULL,
    staz_lata    INTEGER NOT NULL,
    ilosc_sztuk  INTEGER NOT NULL
);

ALTER TABLE scenarzysci ADD CONSTRAINT scenarzysci_pk PRIMARY KEY ( sn_id );

CREATE TABLE sztuki (
    sz_id              INTEGER NOT NULL,
    nazwa              VARCHAR2(20 CHAR) NOT NULL,
    kasa               NUMBER(7, 2) NOT NULL,
    gatunek            VARCHAR2(20 CHAR) NOT NULL,
    sale_sl_id         INTEGER NOT NULL,
    scenariusze_sc_id  INTEGER NOT NULL,
```

```
    repertuary_rp_id   INTEGER
);

CREATE UNIQUE INDEX sztuki__idx ON
    sztuki (
        scenariusze_sc_id
    ASC );

CREATE UNIQUE INDEX sztuki__idx ON
    sztuki (
        sale_sl_id
    ASC );

ALTER TABLE sztuki ADD CONSTRAINT sztuki_pk PRIMARY KEY ( sz_id );

ALTER TABLE bilety
    ADD CONSTRAINT bilety_sztuki_fk FOREIGN KEY ( sztuki_sz_id )
        REFERENCES sztuki ( sz_id );

ALTER TABLE miejsca
    ADD CONSTRAINT miejsca_sale_fk FOREIGN KEY ( sale_sl_id )
        REFERENCES sale ( sl_id );

ALTER TABLE pracownicy
    ADD CONSTRAINT pracownicy_aktorzy_fk FOREIGN KEY ( aktorzy_ak_id )
        REFERENCES aktorzy ( ak_id );

ALTER TABLE pracownicy
    ADD CONSTRAINT pracownicy_rezyserzy_fk FOREIGN KEY ( rezyserzy_rz_id,
                                                         rezyserzy_rezyserzy_id )
        REFERENCES rezyserzy ( rz_id,
                               rezyserzy_id );

ALTER TABLE pracownicy
    ADD CONSTRAINT pracownicy_scenarzysci_fk FOREIGN KEY ( scenarzysci_sn_id )
        REFERENCES scenarzysci ( sn_id );

ALTER TABLE PR_SZ
    ADD CONSTRAINT PR_SZ_pracownicy_fk FOREIGN KEY ( pracownicy_pr_id )
        REFERENCES pracownicy ( pr_id );

ALTER TABLE PR_SZ
    ADD CONSTRAINT PR_SZ_sztuki_fk FOREIGN KEY ( sztuki_sz_id )
        REFERENCES sztuki ( sz_id );

ALTER TABLE SZ_RK
    ADD CONSTRAINT SZ_RK_rekwizyty_fk FOREIGN KEY ( rekwizyty_rk_id )
        REFERENCES rekwizyty ( rk_id );

ALTER TABLE SZ_RK
    ADD CONSTRAINTSZ_RK_sztuki_fk FOREIGN KEY ( sztuki_sz_id )
        REFERENCES sztuki ( sz_id );

ALTER TABLE role
    ADD CONSTRAINT role_sztuki_fk FOREIGN KEY ( sztuki_sz_id )
        REFERENCES sztuki ( sz_id );

ALTER TABLE scenariusze
    ADD CONSTRAINT scenariusze_sztuki_fk FOREIGN KEY ( sztuki_sz_id )
```

```
        REFERENCES sztuki ( sz_id );

ALTER TABLE sztuki
    ADD CONSTRAINT sztuki_repertuary_fk FOREIGN KEY ( repertuary_rp_id )
        REFERENCES repertuary ( rp_id );

ALTER TABLE sztuki
    ADD CONSTRAINT sztuki_sale_fk FOREIGN KEY ( sale_sl_id )
        REFERENCES sale ( sl_id );

ALTER TABLE sztuki
    ADD CONSTRAINT sztuki_scenariusze_fk FOREIGN KEY ( scenariusze_sc_id )
        REFERENCES scenariusze ( sc_id );

CREATE SEQUENCE rezyserzy_rezyserzy_id_seq START WITH 1 NOCACHE ORDER;

CREATE OR REPLACE TRIGGER rezyserzy_rezyserzy_id_trg BEFORE
    INSERT ON rezyserzy
    FOR EACH ROW
    WHEN ( new.rezyserzy_id IS NULL )
BEGIN
    :new.rezyserzy_id := rezyserzy_rezyserzy_id_seq.nextval;
END;
/
```

6. Przypadki użycia bazy danych

6.1. 20 pracowników KTI udaje się do teatru aby obejrzeć spektakl “Kaczuszki”. Chcą napisać sprawozdanie z wycieczki dla studentów aby zachęcić ich do chodzenia do teatru. Daj im podpowiedź, aby dowiedzieli się jaką postać na pewno zobaczą w sztuce oraz oblicz ile pieniędzy skarbnik musi zebrać aby zapłacić za tą wycieczkę, wiedząc że z powodu przybycia tak dużą grupą otrzymują 10% zniżki.

```
SELECT Cena "Cena biletów", r.nazwa "Postać", sz.gatunek "Typ sztuki"
FROM role r
JOIN sztuki sz ON r.sztuki_sz_id =  sz.sz_id
JOIN (SELECT bl_id, sztuki_sz_id, round((20*cena)*0.9,2) Cena
    FROM bilety
    WHERE bl_id = 89
    ) bil
    ON (r.sztuki_sz_id=bil.sztuki_sz_id)
WHERE sz.sz_id = 103;
```

6.2. Magazyn recenzujący sztuki „Ale teatr!” poprosił teatr o udostępnienie wykazu sztuk dramatycznych, w których wystąpili aktorzy dramatyczni. Poproszono o udostępnienie informacji, jaki aktor wcielił się w jaką rolę; dodatkowo pojawiła się potrzeba wskazania, ile stron liczył scenariusz. Teatr chętnie przekaże magazynowi „Ale Teatr!” takie zestawienie, ze względu na rozgłos, jaki może uzyskać wśród czytelników renomowanego magazynu.

```
SELECT Sztuki.Nazwa, Sztuki.Gatunek, Scenariusze.Ilosc_stron,
(Pracownicy.Imie || ' ' || Pracownicy.Nazwisko),
aktorzy.pseudonim, aktorzy.gatunek, role.nazwa, role.typ
```

```
FROM Sztuki
INNER JOIN Scenariusze ON sztuki.scenariusze_sc_id=scenariusze.sc_id
INNER JOIN pr_sz ON sztuki.sz_id=pr_sz.sztuki_sz_id
INNER JOIN Pracownicy ON pracownicy.pr_id=pr_sz.pracownicy_pr_id
INNER JOIN role ON role.sztuki_sz_id=sztuki.sz_id
INNER JOIN aktorzy ON aktorzy.AK_ID=pracownicy.aktorzy_ak_id
WHERE aktorzy.gatunek='dramatyczny' AND Sztuki.Gatunek='dramat';
```

6.3. Zaistniała potrzeba sprawdzenia czy scenariusze dla opłacalnych sztuk nie są zbyt przedawnione. Z tego względu trzeba opublikować zestawienie czterech najbardziej dochodowych sztuk, spośród tych, których scenariusz jest starszy niż średnia „wieku" scenariuszy sztuk w teatrze.

```
SELECT         Sztuki.Nazwa,         Sztuki.kasa,         scenariusze.ilosc_stron,
ROUND(MONTHS_BETWEEN(SYSDATE, scenariusze.data_utworzenia), 0) Jak_stara_mies,
COALESCE(scenariusze.sztuki_sz_id, 0)
FROM scenariusze
INNER JOIN Sztuki
ON scenariusze.sztuki_sz_id=sztuki.sz_id
WHERE ilosc_stron > (SELECT AVG(ilosc_stron)
                     FROM scenariusze
                     WHERE sc_id = scenariusze.sc_id)
ORDER BY kasa DESC
FETCH FIRST 4 ROWS ONLY;
```

6.4. Dla celów analitycznych, potrzebujemy dowiedzieć się, jakie sztuki są opłacalne, tzn. które z nich przyniosły większy dochód („kasę") niż średnia dochodu dla sztuk wystawianych w teatrze; w jakiej cenie średnio sprzedany był bilet na taką sztukę, uwzględniając bilety normalne i ulgowe?

```
SELECT sz.Nazwa, sz.kasa, avg_cena_biletu
FROM Sztuki sz JOIN
    (SELECT sztuki_sz_id, AVG(cena) avg_cena_biletu
    FROM bilety
    GROUP BY sztuki_sz_id) bil
            ON (sz.sz_id=bil.sztuki_sz_id)
WHERE sz.kasa > (SELECT AVG(kasa)
                    FROM Sztuki);
```

6.5. W teatrze miało miejsce włamanie. Domniemamy, że złodzieje bądź wynieśli któryś z rekwizytów, bądź część rekwizytu – np. poczęstowali się spirytusem z butelki. Rekwizyty zostały przeliczone i zważone; wyniki trzeba porównać z ewidencją rekwizytów, by sprawdzić czy teatr nie ucierpiał na wizycie włamywaczy. Trzeba zliczyć, ile rekwizytów powinno być w magazynie i ile powinny ważyć łącznie.

```
SELECT COUNT(*) "Ilosc uzwanych rk", SUM(waga) "Suma wagi"
```

```
FROM rekwizyty rk
WHERE rk_id IN
    (SELECT rekwizyty_rk_id
    FROM sz_rk
    WHERE rk.rk_id = sz_rk.rekwizyty_rk_id);
```

6.6. Jakie sztuki były wystawiane w teatrze w czwartym kwartale roku 2022? Do kiedy obowiązywał ten repertuar?

```
SELECT    (TO_CHAR(data_rozpoczecia,    'YYYY-MON-DD')    ||    '    do    '    ||
(TO_CHAR(data_zakonczenia, 'YYYY-MON-DD'))) Kiedy_w_repertuarze,
sz.Nazwa
FROM repertuary JOIN sztuki sz
ON repertuary.rp_id=sz.repertuary_rp_id
WHERE data_rozpoczecia='22/10/01';
```

6.7. Czy dwie najbardziej dochodowe sale są dostosowane dla potrzeb osób z niepełnosprawnościami?

```
SELECT s.nazwa, SUM(sz.kasa) "LACZNA KASA DLA SALI", s.czy_dla_niepelnosprawnych
    FROM Sztuki sz
        JOIN Sale s
            ON sz.sale_sl_id = s.sl_id
        GROUP BY s.nazwa, s.czy_dla_niepelnosprawnych
            ORDER BY SUM(sz.kasa) DESC
                FETCH FIRST 2 ROWS ONLY;
```

6.8. Teatr zamierza ruszyć z reklamą sztuk w szkołach. W tym celu należy sprawdzić, na które sztuki zakupiono najwięcej biletów ulgowych.

```
SELECT sz.nazwa, COUNT(b.czy_ulgowy) "SPRZEDANE BILETY ULGOWE"
    FROM sztuki sz
        JOIN bilety b
            ON b.sztuki_sz_id = sz.sz_id
        WHERE b.czy_ulgowy = 1
            GROUP BY sz.nazwa;
```

7. Pozostałe obiekty bazy danych

7.1. Tabela asocjacyjna dla rekwizytów i sztuki

```
insert into SZ_RK(SZTUKI_SZ_ID, REKWIZYTY_RK_ID)
(SELECT    SZTUKI.SZ_ID,    REKWIZYTY.RK_ID    FROM    SZTUKI,REKWIZYTY    WHERE
SZTUKI.NAZWA='Wesele' AND REKWIZYTY.NAZWA='slomiany posag');

insert into SZ_RK(SZTUKI_SZ_ID, REKWIZYTY_RK_ID)
(SELECT    SZTUKI.SZ_ID,    REKWIZYTY.RK_ID    FROM    SZTUKI,REKWIZYTY    WHERE
SZTUKI.NAZWA='Kogel Mogel' AND REKWIZYTY.NAZWA='butelka spirytusu');
```

```
insert into SZ_RK(SZTUKI_SZ_ID, REKWIZYTY_RK_ID)
(SELECT    SZTUKI.SZ_ID,    REKWIZYTY.RK_ID    FROM    SZTUKI,REKWIZYTY    WHERE
SZTUKI.NAZWA='Kaczuszki' AND REKWIZYTY.NAZWA='gumowy nos');

insert into SZ_RK(SZTUKI_SZ_ID, REKWIZYTY_RK_ID)
(SELECT    SZTUKI.SZ_ID,    REKWIZYTY.RK_ID    FROM    SZTUKI,REKWIZYTY    WHERE
SZTUKI.NAZWA='Makbet' AND REKWIZYTY.NAZWA='suknia Lady Makbet');

insert into SZ_RK(SZTUKI_SZ_ID, REKWIZYTY_RK_ID)
(SELECT    SZTUKI.SZ_ID,    REKWIZYTY.RK_ID    FROM    SZTUKI,REKWIZYTY    WHERE
SZTUKI.NAZWA='Zemsta' AND REKWIZYTY.NAZWA='cegla');

insert into SZ_RK(SZTUKI_SZ_ID, REKWIZYTY_RK_ID)
(SELECT    SZTUKI.SZ_ID,    REKWIZYTY.RK_ID    FROM    SZTUKI,REKWIZYTY    WHERE
SZTUKI.NAZWA='Kaczuszki' AND REKWIZYTY.NAZWA='zolty plaszcz');

insert into SZ_RK(SZTUKI_SZ_ID, REKWIZYTY_RK_ID)
(SELECT    SZTUKI.SZ_ID,    REKWIZYTY.RK_ID    FROM    SZTUKI,REKWIZYTY    WHERE
SZTUKI.NAZWA='Makbet' AND REKWIZYTY.NAZWA='miecz');

insert into SZ_RK(SZTUKI_SZ_ID, REKWIZYTY_RK_ID)
(SELECT    SZTUKI.SZ_ID,    REKWIZYTY.RK_ID    FROM    SZTUKI,REKWIZYTY    WHERE
SZTUKI.NAZWA='Zemsta' AND REKWIZYTY.NAZWA='krokodyl');
```

7.2. Tabela asocjacyjna dla pracowników i sztuk

```
INSERT INTO PR_SZ(PRACOWNICY_PR_ID, SZTUKI_SZ_ID)
(SELECT    PRACOWNICY.PR_ID,    SZTUKI.SZ_ID    FROM    PRACOWNICY,SZTUKI    WHERE
PRACOWNICY.NAZWISKO='Kwiatkowski' AND SZTUKI.NAZWA='Kogel Mogel');

INSERT INTO PR_SZ(PRACOWNICY_PR_ID, SZTUKI_SZ_ID)
(SELECT    PRACOWNICY.PR_ID,    SZTUKI.SZ_ID    FROM    PRACOWNICY,SZTUKI    WHERE
PRACOWNICY.NAZWISKO='Smolarek' AND SZTUKI.NAZWA='Zemsta');

INSERT INTO PR_SZ(PRACOWNICY_PR_ID, SZTUKI_SZ_ID)
(SELECT    PRACOWNICY.PR_ID,    SZTUKI.SZ_ID    FROM    PRACOWNICY,SZTUKI    WHERE
PRACOWNICY.NAZWISKO='Nowaczyk' AND SZTUKI.NAZWA='Wesele');

INSERT INTO PR_SZ(PRACOWNICY_PR_ID, SZTUKI_SZ_ID)
(SELECT    PRACOWNICY.PR_ID,    SZTUKI.SZ_ID    FROM    PRACOWNICY,SZTUKI    WHERE
PRACOWNICY.NAZWISKO='Mostowiak' AND SZTUKI.NAZWA='Kaczuszki');
```

7.3. Generowanie ID poprzez sekwencjonera w tabeli REKWIZYTY

```
CREATE SEQUENCE rk_id_seq
INCREMENT BY 1
START WITH 39
MAXVALUE 200
NOCACHE
NOCYCLE;

INSERT INTO rekwizyty VALUES (rk_id_seq.NEXTVAL, 'pistolet', 0.86, 1);
```

```
INSERT INTO rekwizyty VALUES (rk_id_seq.NEXTVAL, 'miecz', 1.3, 1);

INSERT INTO rekwizyty VALUES (rk_id_seq.NEXTVAL, 'zolty plaszcz', 0.35, 0);

INSERT INTO rekwizyty VALUES (rk_id_seq.NEXTVAL, 'krokodyl', 523.5, 1);

INSERT INTO rekwizyty VALUES (rk_id_seq.NEXTVAL, 'rower', 12.5, 0);
```

7.4. Generowanie ID poprzez sekwencjonera w tabeli SCENARIUSZE

```
CREATE SEQUENCE sc_id_seq
INCREMENT BY 1
START WITH 6
MAXVALUE 50
NOCACHE
NOCYCLE;

INSERT INTO scenariusze VALUES (sc_id_seq.NEXTVAL, 40, to_date('19/06/24',
'yy/mm/dd'), NULL);

INSERT INTO scenariusze VALUES (sc_id_seq.NEXTVAL, 520, to_date('19/02/03',
'yy/mm/dd'), NULL);

INSERT INTO scenariusze VALUES (sc_id_seq.NEXTVAL, 134, to_date('20/01/10',
'yy/mm/dd'), NULL);

INSERT INTO scenariusze VALUES (sc_id_seq.NEXTVAL, 83, to_date('20/12/05',
'yy/mm/dd'), NULL);
```

7.5. Generowanie ID poprzez sekwencjonera w tabeli ROLE

```
CREATE SEQUENCE rl_id_seq
INCREMENT BY 1
START WITH 213
MAXVALUE 300
NOCACHE
NOCYCLE;

INSERT INTO role (rl_id, nazwa, typ, sztuki_sz_id) VALUES (rl_id_seq.NEXTVAL,
'Chochol', 'dramatyczny', 101);
INSERT INTO role (rl_id, nazwa, typ, sztuki_sz_id) VALUES (rl_id_seq.NEXTVAL,
'Dziennikarz', 'dramatyczny', 101);
INSERT INTO role (rl_id, nazwa, typ, sztuki_sz_id) VALUES (rl_id_seq.NEXTVAL,
'Pani Solska', 'komediowy', 102);
INSERT INTO role (rl_id, nazwa, typ, sztuki_sz_id) VALUES (rl_id_seq.NEXTVAL,
'Kaczka obrazalska', 'bajkowy', 103);
INSERT INTO role (rl_id, nazwa, typ, sztuki_sz_id) VALUES (rl_id_seq.NEXTVAL,
'Naczelnik', 'komediowy', 102);
```

7.6. Perspektywy

7.6.1

```
CREATE VIEW WYCIECZKA_KTI AS
SELECT Cena "Cena biletów", r.nazwa "Postać", sz.gatunek "Typ sztuki"
FROM role r
JOIN sztuki sz ON r.sztuki_sz_id =  sz.sz_id
JOIN (SELECT bl_id, sztuki_sz_id, round((20*cena)*0.9,2) Cena
    FROM bilety
    WHERE bl_id = 89
    ) bil
    ON (r.sztuki_sz_id=bil.sztuki_sz_id)
WHERE sz.sz_id = 103;
```

7.6.2

```
CREATE VIEW SZTUKI_ORAZ_AKTORZY_DRAMATY AS
SELECT Sztuki.Nazwa, Sztuki.Gatunek, Scenariusze.Ilosc_stron,
(Pracownicy.Imie || ' ' || Pracownicy.Nazwisko),
aktorzy.pseudonim, aktorzy.gatunek, role.nazwa, role.typ
FROM Sztuki
INNER JOIN Scenariusze ON sztuki.scenariusze_sc_id=scenariusze.sc_id
INNER JOIN pr_sz ON sztuki.sz_id=pr_sz.sztuki_sz_id
INNER JOIN Pracownicy ON pracownicy.pr_id=pr_sz.pracownicy_pr_id
INNER JOIN role ON role.sztuki_sz_id=sztuki.sz_id
INNER JOIN aktorzy ON aktorzy.AK_ID=pracownicy.aktorzy_ak_id
WHERE aktorzy.gatunek='dramatyczny' AND Sztuki.Gatunek='dramat';
```

7.6.3

```
CREATE VIEW PODSUMOWANIE_DLUGICH_SZTUK AS
SELECT         Sztuki.Nazwa,         Sztuki.kasa,         scenariusze.ilosc_stron,
ROUND(MONTHS_BETWEEN(SYSDATE, scenariusze.data_utworzenia), 0) Jak_stara_mies,
COALESCE(scenariusze.sztuki_sz_id, 0)
FROM scenariusze
INNER JOIN Sztuki
ON scenariusze.sztuki_sz_id=sztuki.sz_id
WHERE ilosc_stron > (SELECT AVG(ilosc_stron)
                     FROM scenariusze
                     WHERE sc_id = scenariusze.sc_id)
ORDER BY kasa DESC
FETCH FIRST 4 ROWS ONLY;
```

7.6.4

```
CREATE VIEW OPLACALNE_SZTUKI_AVG_CENA AS
SELECT sz.Nazwa, sz.kasa, avg_cena_biletu
FROM Sztuki sz JOIN
    (SELECT sztuki_sz_id, AVG(cena) avg_cena_biletu
    FROM bilety
    GROUP BY sztuki_sz_id) bil
            ON (sz.sz_id=bil.sztuki_sz_id)
WHERE sz.kasa > (SELECT AVG(kasa)
                   FROM Sztuki);
```

7.6.5

```
CREATE VIEW SUM_WAGA_UZYWANYCH_RK AS
SELECT COUNT(*) "Ilosc uzwanych rk", SUM(waga) "Suma wagi"
FROM rekwizyty rk
WHERE rk_id IN
    (SELECT rekwizyty_rk_id
    FROM sz_rk
    WHERE rk.rk_id = sz_rk.rekwizyty_rk_id);
```

7.6.6

```
CREATE VIEW REPERTUAR_OD_PAZ2022 AS
SELECT    (TO_CHAR(data_rozpoczecia,    'YYYY-MON-DD')    ||    '    do    '    ||
(TO_CHAR(data_zakonczenia, 'YYYY-MON-DD'))) Kiedy_w_repertuarze,
sz.Nazwa
FROM repertuary JOIN sztuki sz
ON repertuary.rp_id=sz.repertuary_rp_id
WHERE data_rozpoczecia='22/10/01';
```

7.6.7

```
CREATE VIEW TOP$_SALE_A_NIEPELNOSPRAWNI AS
SELECT s.nazwa, SUM(sz.kasa) "LACZNA KASA DLA SALI", s.czy_dla_niepelnosprawnych
    FROM Sztuki sz
        JOIN Sale s
            ON sz.sale_sl_id = s.sl_id
        GROUP BY s.nazwa, s.czy_dla_niepelnosprawnych
            ORDER BY SUM(sz.kasa) DESC
                FETCH FIRST 2 ROWS ONLY;
```

7.6.8

```
CREATE VIEW TOP_SZTUKI_W_ULGOWYCH AS
SELECT sz.nazwa, COUNT(b.czy_ulgowy) "SPRZEDANE BILETY ULGOWE"
    FROM sztuki sz
        JOIN bilety b
            ON b.sztuki_sz_id = sz.sz_id
        WHERE b.czy_ulgowy = 1
            GROUP BY sz.nazwa;
```