REST API

Data z pokladen jsou pravidelně synchronizována do cloudu a zpřístupněna přes REST API.

> API je v současné době poskytováno v omezeném režimu. Do budoucna plánujeme expiraci autorizačního tokenu. Pokud chcete získávat kontinuáně data a synchronizovat systémy **doporučujeme používat Webhooky**. Jedná se o efektivnější způsob.

Autorizace požadavku

Autorizace probíhá pomocí API Tokenu, který naleznete v **Backoffice - Nastavení - Systém.**

Autorizační klíč je v HTTP hlavičce `Authorization: A|xxxxx`

```
curl -XGET 'https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/products' \
 -H 'Authorization: A|kLgT.........' \
 -H 'Content-Type: application/json'
```

POST přííklad:

```
curl -XPOST 'https://m6vadtaz1h.execute-api.eu-west-
1.amazonaws.com/prod/data/reports/generic/order_history' \
 -H 'Authorization: A|kLgT.........' \
 -H 'Content-Type: application/json' \
 --data-raw
'{"date_start":1639436400000,"date_end":1642028399999,"timezone":"Europe/Prague","id_cash_regi
ster":953221254382092,"id_shop":492696696397774}' \
```

Konvence

- Časy jsou reprezentovány v unix timestamp v millisekundách. Např. 1639436400000
- Monetární hodnoty (tržba, cena, atd) jsou vždy celé číslo, vynásobené 1000. Např. cena 23.50 je v JSON dokumentu jako 23500.
- Autorizační token nemá expiraci, ale do budoucna plánujeme že token bude muset být vyměněn za nový
- Do User Agent prosím vložte svoji identifikaci případně kontakt, abychom měli kontakt na vývojáře
- Pokud navrácený objekt obsahuje položku, která začíná na `__` tak se jedná o tzv. hydrataci - objekt je automaticky obohacen o child objekty, aby vývojář nemusel provádět další dotazy na API.
- Převod id na čas lze provést pomocí funkce: `new Date((id / 32768) + 1440000000000);`

Společné pole

- `id_c` - globální ID zákazníka
- `_v` - verze objektu
- `_d` - indikuje jestli byl záznam označen jako smazaný
- `_t` - název tabulky
- `pgx` - složený interní primární klíč - ignorujte, pokud se vyskytuje
- pole s prefixem `date_` jsou Unix UTC čas v millisekundách
- pole s prefixem `id_` jsou relace a odkaz na záznam v jiné tabulce. Např `id_shop` odkazuje na id obchodu v tabulce shops.

Funkce pro práci s id

Id je číslo složené z času, id tabulky a náhodného suffixu. Příklad výpočtu:

```
// Javascript
const EPOCH = 1440000000000;

// Vrati objekt s datem a id tabulky
function getDateFromId(id) {
  return {
    dt: new Date((id / 32768) + EPOCH),
    tableId: (id / 512) & 0x3f
  }
}

// Generuje id pro konkretni tabulku - pouzijte 0 pro referenci
exports.getRandomRowId = function (tableId) {
    let ts = new Date().getTime() - EPOCH;
    let randid = Math.floor(Math.random() * 511);
    ts = (ts * 64);
    ts = ts + tableId;
    let t = (ts * 512) + (randid % 512);
    while (t % 10 !== 0) {
        t = t + 2; // Resi chybu s 53bit number v Javascriptu
    }
    return t;
}
```

```
// C#
const long EPOCH = 1440000000000; // Matches the JavaScript EPOCH value

public static long GetRandomRowId(int tableId) {
   long ts = DateTimeOffset.UtcNow.ToUnixTimeMilliseconds() - EPOCH;
   int randid = new Random().Next(512); // Generates a random number between 0 and 511
   ts = (ts * 64);
   ts = ts + tableId;
   long t = (ts * 512) + (randid % 512);

   t = t - (t % 10);
   return t;
 }
```

Seznam endpoint?

Následující endpointy podporují metody **GET** (získání seznamů dat json array) i **POST** (vytvoření nebo aktualizace jednoho objektu). Metoda POST podporuje i zaslání více objektů v JSON Array. Lze využít pro dávkové aktualizace více objektů.

Base URL: `https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/`

- products
- customers
- product_price_matrix
- shops
- cash_registers
- employees
- park_locations
- shifts
- stock_history
- stockup
- transactions
- warehouses
- *order_details a orders - podporuje GET i POST - nedoporučujeme vytvářet nové objekty pomocí POST, proč?*

Základní princip stránkování

API vrací výsledky v stránkách (pages). Každá odpověď obsahuje maximálně `limit` záznamů. Pro získání dalších záznamů použijte parametr `id_start`.

Stránkování je založeno na **ID záznamu** (BigInt), nikoliv na čase. ID je Unix timestamp v milisekundách jako BigInt, takže vyšší ID = novější záznam.

Parametry stránkování

Parametr	Typ	Popis	Výchozí hodnota
`limit`	integer	Maximální počet záznamů na stránku (1-400)	100
`id_start`	string	ID posledního zobrazeného záznamu pro další stránku	`0`
`direction`	string	`forward` (vzestupně) nebo `backward` (sestupně)	`backward` pro tabulky orders, transactions, shifts
`version_start`	integer	Filtruje záznamy s verzí `_v` >= hodnota	`0` (bez filtru)

Sm?ry stránkování

Backward (výchozí pro orders, transactions, shifts)

Vrací **nejnovější záznamy první** (sestupně podle data vytvoření).

```
GET /data/orders?limit=100
→ vrací 100 nejnovějších objednávek
```

Forward

Vrací **nejstarší záznamy první** (vzestupně podle data vvoření).

```
GET /data/orders?limit=100&direction=forward
→ vrací 100 nejstarších objednávek
```

Jak na stránkování v praxi

cURL

```
# První stránka
curl "/data/orders?limit=400&direction=forward" \
  -H "Authorization: Bearer $TOKEN"

# Další stránka (použijte ID posledního záznamu z předchozí odpovědi)
curl "/data/orders?limit=400&id_start=1115405820218937O&direction=forward" \
```

```
  -H "Authorization: Bearer $TOKEN"
```

Filtrování podle verze

Parametr `version_start` umožňuje získat pouze záznamy aktualizované po určitém čase:

```
GET /data/orders?version_start=1780600680635
→ vrací pouze objednávky s _v >= 1780600680635 (timestamp v ms)
```

P?íklad kompletního cyklu

Máte 1500 objednávek, limit = 400:

Stránka	id_start	Vrácených	Kód
1	`0`	400	`last_id = 11154058202189370`
2	`11154058202189370`	400	`last_id = 11139738857705400`
3	`11139738857705400`	400	`last_id = 11131523820818290`
4	`11131523820818290`	300	← poslední stránka (300 < 400)

D?ležitá upozorn?ní

1. **ID jako string**: `id` v odpovědi je BigInt, může být velmi velké číslo. Při předávání jako `id_start` ho převeďte na string, aby nedošlo ke ztrátě přesnosti.
2. **Nikdy neměňte pořadí**: Pokud stránkujete `forward`, pokračujte vždy `forward`. Míchání směrů může vést k přeskočeným nebo duplicitním záznamům.
3. **Timestamp-based ID**: ID objednávek je založeno na timestampu. Novější objednávky mají vyšší ID. To znamená, že:
   - `direction=backward`: první objednávka = nejnovější
   - `direction=forward`: první objednávka = nejstarší
4. **Limit 2000**: Přestože můžete nastavit `limit` až do 400, DynamoDB dotaz má interně limit 2000 záznamů na jedno volání. Pro větší datasety použijte stránkování.
5. **Nikdy neskipujte stránky**: Vždy iterujte sekvenčně. Přeskočení na "další stránku" bez procházení předchozích může vést ke ztrátě dat.

Chybové stavy

Kód	Popis	Řešení

400	Neplatný `id_start`	Ověřte, že `id_start` je platné ID nebo `0`
429	Rate limit	Snižte frekvenci požadavků
500	Interní chyba	Opakujte požadavek

GET products

Vrací JSON Array se seznamem produktů. Položky které jsou smazané, mají nastaveno `_visible = false`

CURL příklad

```
curl 'https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/products' \
-H 'Authorization: A|......'
```

Odpov??


```
[
  {
    "unit": 0,
    "date_updated": 1641499263724,
    "attributes_bitmask": 8208,
    "__price_matrix_model": {
      "date_starts": 1640070857424,
      "unit_price_base_tax_incl": 550,
      "unit_price_regular_tax_excl": 454546,
      "unit_price_c_tax_excl": 0,
      "credits_deduct": 0,
      "hour_bitmask": 0,
      "points_deduct": 0,
      "reduction_amount_a_tax_excl": 0,
      "id_exec_rule": 0,
      "id_product": 6555921856010421,
      "unit_price_b_tax_excl": 0,
      "dow_bitmask": 0,
      "price_bitmask": 0,
      "unit_price_base_tax_excl": 454.5455,
      "id_shop": 0,
      "currency": "CZK",
      "unit_price_a_tax_excl": 454546,
      "id": 6555922259744395,
```

```
      "id_warehouse": 953221254427235,
      "_t": "product_price_matrix",
      "_v": 1641499263728
    },
    "condition_type": 0,
    "name": "Cesta",
    "id_category": 5723345161520367,
    "node_sort": -6680027574804497000,
    "inventory_management": false,
    "id": 6555921856010421,
    "id_shop": 0,
    "color": 0,
    "visible": true,
    "tax_pst_rate": 1.21,
    "item_type": 0,
    "icon_code": 0,
    "quantity_minimal": 1,
    "quantity_multiple": 1,
    "id_c": 111111,
    "_t": "products",
    "_v": 1641499263746
  }
]
```


GET product/{id}

Vrací jednotlivý produkt podle ID objektu. Pro ceny volejte get-price-by-product-id/

```
https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/products/{id}
```

Odpov??


```
{
  "unit": 0,
  "pgx": "111:products",
  "description_short": "Vydařená kombinace rumu a griotky",
  "date_updated": 1647012279153,
  "attributes_bitmask": 8240,
  "sale_group_type": 0,
  "condition_type": 1,
  "name": "Čert",
```

```
  "id_tax_rules_group": 0,
  "id_category": 6783340285035904,
  "node_sort": 3752073357922477000,
  "accounting_group": 600000,
  "warranty_length": 0,
  "inventory_management": true,
  "inventory_type": 0,
  "id": 6783372795320333,
  "id_shop": 0,
  "tax_eco": 1,
  "color": 0,
  "visible": true,
  "_d": 0,
  "tax_luxury": 1,
  "tax_pst_rate": 1.21,
  "item_type": 0,
  "additional_shipping_cost": 0,
  "icon_code": 0,
  "warranty_type": 0,
  "_t": "products",
  "quantity_minimal": 1,
  "_v": 1681918519463,
  "quantity_multiple": 1,
  "id_c": 111111,
  "id_exec_rule": 0
}
```


POST product

Vytvoří nový produkt. Použijte id produktu `0` pro nový objekt, nebo id existujícího produktu pro aktualizaci objektu. Pro informace o formátu objektu se podívejte do definice zod, viz níže.

Poznámky:

- id_shop=0 indikuje ze položka není omezena na žádnou pobočku
- attributes_bitmask= je bitmaska s atributy. Používají se bitové operace, hodnota se ukládá jako int

```
curl 'https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/products' \
  -H 'Accept: application/json' \
  -H 'Authorization: A|....' \
```

```
-H 'Content-Type: application/json' \
-d '{
  "deleted": 0,
  "id": 0, // Zero or omit to create new record
  "_t": "products",
  "image_dimension": null,
  "notes_quick": "Sample quick notes",
  "quantity_multiple": 1,
  "quantity_minimal": 1,
  "additional_shipping_cost": 0,
  "date_updated": 1678886400000,  // Example timestamp (replace as needed)
  "inventory_management": false,
  "color": 0,
  "item_type": 0,
  "id_category": 0,
  "tax_luxury": 1.000,
  "ean13": "1234567890123",
  "id_tax_rules_group": 0,
  "name_alternative": "Alt Product Name",
  "sku": "PROD-SKU-001",
  "tax_eco": 1.000000,
  "barcode": "9876543210987",
  "height": 10.5,
  "visible": true,
  "image_url": "https://example.com/image.jpg",
  "warranty_length": 12,
  "condition_type": 0,
  "tax_pst_rate": 1.000,
  "upc": "012345678901",
  "weight": 1.2,
  "icon_code": 0,
  "details_url": "https://example.com/details",
  "warranty_type": 0,
  "tags": "sample,product",
  "description_short": "Short product description",
  "unit": 0,
  "id_exec_rule": 0,
  "depth": 5.0,
  "sale_group_type": 0,
  "id_global_product_code": null,
```

```
    "accounting_group": 600000,
    "name": "Sample Product Name",
    "width": 7.8,
    "node_sort": 0,
    "attributes_bitmask": 0,
    "inventory_type": 0,
    "internal_extra": "Internal notes",
    "id_shop": 0
  }'
```

GET get-price-by-product-id/{id_product}

Vrací ceny pro konkrétní produkt v ceníku podle id produktu.

```
https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/products/get-price-by-product-id/{id_product}
```

GET product_price_matrix

Ceny k produktům. Je nutné napárovat pomocí id_product

```
https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/product_price_matrix
```

Odpov??


```
[
  {
    "currency": "CZK",
    "unit_price_base_tax_excl": 30,
    "id_warehouse": 4438373662962620,
    "_d": 1,
    "dow_bitmask": 0,
    "credits_deduct": 0,
    "hour_bitmask": 0,
    "unit_price_base_tax_incl": 30,
    "id_product": 4438453021022482,
    "unit_price_regular_tax_excl": 30000,
    "points_deduct": 0,
```

```
    "date_starts": 1575450836824,
    "price_bitmask": 0,
    "_t": "product_price_matrix",
    "unit_price_a_tax_excl": 30000,
    "_v": 1633950008092,
    "id": 4438454091814580,
    "reduction_amount_a_tax_excl": 0,
    "id_shop": 0,
    "id_c": 1141932
   }
  ]
```


GET categories

Seznam kategorií v hiearchické struktuře

```
https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/categories
```

Odpov??


```
[
  {
    "visible": true,
    "_d": 0,
    "name": "Cakes >",
    "category_bitmask": 0,
    "icon_code": 5,
    "node_sort": 5000000,
    "id_category_sort_after": 0,
    "_t": "categories",
    "_v": 1672597351272,
    "id_category_parent": 0,
    "id": 1,
    "id_shop": 0,
    "color": 17,
    "id_c": 111111,
    "children": [
      {
        "visible": false,
        "name": "test",
        "category_bitmask": 0,
```

```
        "margin_minimal_rate": null,
        "icon_code": 4,
        "node_sort": 5000000,
        "id_category_sort_after": 0,
        "_t": "categories",
        "_v": 1640029337604,
        "id_category_parent": 1,
        "id": 1067887041152062,
        "id_shop": 0,
        "margin": null,
        "color": 5,
        "tags": null,
        "id_c": 111111,
        "children": []
      }
    ]
  }
]
```


GET shops

Seznam poboček (shops)

```
https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/shops
```

Odpov??


```
[
  {
    "visible": true,
    "currency": "CZK",
    "_d": 0,
    "name": "Shop 83",
    "location_name": "City 44",
    "_t": "shops",
    "_v": 1677240349376,
    "id": 492696696391111,
    "bitmask": 0,
    "id_c": 111111
  }
]
```

GET cash_registers

Seznam poboček. Více o struktuře a závislosti na obchodech v manuálu zde.

```
https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/cash_registers
```

Odpov??


```
[
  {
    "visible": true,
    "name": "Point of Sale 233",
    "location_name": "Brno",
    "_t": "cash_registers",
    "_v": 1640802641284,
    "cash_register_type": 11,
    "id": 49269669639111,
    "id_shop": 492696696397774,
    "bitmask": 0,
    "id_c": 11111
  }
]
```


GET customers

Seznam zákazníků

```
curl 'https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/customers' \
 -H 'Authorization: A|ADq
```

Odpov??


```
[
  {
    "customer_type": 0,
    "credit_account": null,
    "internal_extra": null,
    "price_group": "A",
    "zip_code": "40801",
    "date_updated": null,
    "email": null,
    "city": "RUMBURK",
```

```
    "bank_account": null,
    "date_birthday": null,
    "company": "Test s.r.o.",
    "geohash": null,
    "firstname": null,
    "company_ico": "3332650",
    "id": 3665795354036044,
    "color": 0,
    "bitmask": 0,
    "tags": "csv-import-91",
    "barcode": null,
    "visible": true,
    "id_discount_group": null,
    "lastname": null,
    "company_dic": "CZ33333",
    "internal_json": null,
    "_t": "customers",
    "_v": 1551875000154,
    "date_expires": null,
    "phone_number": null,
    "note_internal": null,
    "country_code": "CZ",
    "note_external": null,
    "id_c": "11111",
    "street": "17.8"
  }
]
```


GET customer/{id}

Jednotlivý záznam pro zákazníka podle ID.

```
https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/customers/{id}
```

```
curl 'https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/customers/12344566' \
 -H 'Authorization: A|ADqsEX...
```

GET orders

Vrací seznam účtenek (JSON Array), seřazené od nejnovějších po nejstarší.

Poznámky:

- Pokud je doklad zrušený nebo stornovaný, pole `date_canceled` obsahuje čas zrušení (not-null)
- `reduction_percent` indikuje slevu v procentech poskytnutou zakaznikovi vynasobene 1000, uvedené jako celé číslo (integer). Tzn. hodnota 500 znamená, že byla poskytnuta sleva 0.5%

Parametry

- id_start - id záznamu od kterého se má stránkovat. výchozí hodnota 0 stránkuje od nejnovejší účtenky
- version_start - unix epoch timestamp (ms) filtr. Hodnota 0 znamená že se hodnoty nebudou filtrovat.
- limit - omezit počet záznamů, výchozí 250, maximum 2000. Velková velikost navráceného JSON nemůže přesáhnout 6MB

```
https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/orders?version_start=0&id_start={id_order}
```

GET orders/latest/{id_cash_register}

Posledních 40 orders z konkrétní pokladny podle id_cash_register. Metoda nepodporuje stránkování. Doporučujeme na získávání aktuálních objednávek v pravidelných intervalech pro systémy, které chtějí získávat online přehled o uskutečnených pohybech.

Query string parametry:

- id_payment (optional) - ID forma platby. Např 221: hotovost, 222: kreditní parta, 238: sumup

```
https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/orders/latest/{id_cash_register}?id_payment={id_payment}
```

Odpov??


```
[
  {
    "total_discounts_tax_excl": 0,
    "currency": "CZK",
    "pgx": "1141932:orders:4438373662917304",
    "total_points": 0,
    "id_employee": 0,
    "date_closed": 1642005467495,
```

```
 "total_credits": 0,
 "id_payment": 222,
 "total_paid_tax_excl": 50000,
 "total_shipping_tax_excl": 0,
 "total_tax_service": 0,
 "total_paid_real": 50000,
 "conversion_rate": 1,
 "_u_dyn": 1642005490213,
 "total_discounts_tax_incl": 0,
 "geohash": "u2ugrgc",
 "date_collected": 1642005467495,
 "total_tip": 0,
 "id_cash_register": 4438373662917304,
 "id": 6619285123010267,
 "dine_in": true,
 "id_shop": 4438373662924485,
 "total_products": 1,
 "total_tax_gst": 0,
 "total_profit_tax_excl": 0,
 "order_serial_number": 130,
 "summary": "1x Pure bar",
 "id_park_location": 0,
 "total_wrapping_tax_incl": 0,
 "_d": 0,
 "total_tax_eco": 0,
 "total_tax_pst": 0,
 "total_shipping_tax_incl": 0,
 "id_employee_served": 0,
 "date_paid": 1642005467185,
 "id_shift": 6618245790277082,
 "total_profit_tax_incl": 0,
 "date_tax_reported": 1642005467495,
 "_t": "orders",
 "total_paid_tax_incl": 50000,
 "_v": 1642005467521,
 "total_wrapping_tax_excl": 0,
 "total_tax_luxury": 0,
 "invoice_number": 202012274,
 "id_c": 1141932
}
```

```
]
```

GET order/{id_cash_register}/{id_order}

Umožní získat účtenku/doklad včetně položek v poli **__order_details**

```
https://m6vadtaz1h.execute-api.eu-west-
1.amazonaws.com/prod/data/orders_details/{id_cash_register}/{id_order}
```


```
{
  "total_discounts_tax_excl": 0,
  "currency": "CZK",
  "pgx": "1173665:orders:7694791899251239",
  "total_points": 0,
  "id_employee": 1,
  "date_closed": 1684414596504,
  "total_credits": 0,
  "id_payment": 222,
  "total_paid_tax_excl": 56522,
  "total_shipping_tax_excl": 0,
  "total_tax_service": 0,
  "date_fiscalized": 0,
  "total_paid_real": 65000,
  "conversion_rate": 1,
  "total_discounts_tax_incl": 0,
  "date_collected": 1684414596504,
  "total_tip": 0,
  "id_cash_register": 7694791899251000,
  "id": 8008974168592088,
  "dine_in": true,
  "id_shop": 7694791899258554,
  "total_products": 2,
  "total_tax_gst": 0,
  "total_profit_tax_excl": 0,
  "order_serial_number": 320,
  "id_park_location": 0,
  "total_wrapping_tax_incl": 0,
  "_d": 0,
  "total_tax_eco": 0,
  "total_tax_pst": 8478,
```

```
"total_shipping_tax_incl": 0,
"id_employee_served": 1,
"date_paid": 1684414596504,
"id_shift": 7962837799869694,
"total_profit_tax_incl": 0,
"date_tax_reported": 1684414596504,
"_t": "orders",
"total_paid_tax_incl": 65000,
"_v": 1684414596517,
"total_wrapping_tax_excl": 0,
"total_tax_luxury": 0,
"invoice_number": 202305288,
"id_c": 11111,
"date_canceled": null,
"__order_details": [
  {
    "product_item_type": 0,
    "currency": "CZK",
    "pgx": "1173665:order_details:7694791899251239",
    "total_points": 0,
    "id_warehouse": 0,
    "id_employee": 1,
    "date_updated": 1684414596517,
    "total_credits": 0,
    "id_payment": 222,
    "id_product": 7833117185743150,
    "_u_dyn": 1684414622686,
    "reduction_amount_tax_incl": 0,
    "id_category": 7694792007615808,
    "id_order": 8008974168592088,
    "product_name_alternative": ":;en:Cappuccino",
    "product_price_billed_tax_incl": 35000,
    "id_order_details_parent": 1,
    "id_cash_register": 7694791899251000,
    "id": 8008976796814560,
    "id_shop": 7694791899258554,
    "product_quantity": 1,
    "product_bitmask": 0,
    "product_name": "Cappuccino",
    "_d": 0,
```

```
      "total_tax_eco": 0,
      "total_tax_pst": 4565,
      "reduction_amount_tax_excl": 0,
      "product_price_billed_tax_excl": 30435,
      "product_unit": 0,
      "product_price_original_tax_excl": 30435,
      "tax_pst_rate": 1.15,
      "date_paid": 1684414596517,
      "id_shift": 7962837799869694,
      "_t": "order_details",
      "_v": 1684414596517,
      "order_number": 320,
      "total_tax_luxury": 0,
      "reduction_percent": 0,
      "id_c": 11111,
      "date_voided": null,
      "date_canceled": null
    }
  ]
}
```


GET order_details

Vrací položky účtenek. Jedná se o child kolekci objektu orders.

```
https://m6vadtaz1h.execute-api.eu-west-
1.amazonaws.com/prod/data/order_details?version_start=0&id_start={id_order_details}
```

POST warehouse_status

Aktuální stav skladu. Je nutné zaslat JSON request s id_warehouse který určuje ID skladu, pro který chcete report.

```
curl 'https://m6vadtaz1h.execute-api.eu-west-
1.amazonaws.com/prod/data/reports/generic/warehouse_status' \
  -H 'Accept: application/json' \
  -H 'Authorization: A|....' \
  -H 'Content-Type: application/json' \
  --data-raw '{"id_warehouse":953221250000000}'
```

Odpov??

```
{
  "data": [
    {
      "id_c": 111111,
      "id_warehouse": 953221254427235,
      "wname": "WAREHOUSE MAIN 21",
      "id_product": 5128991459121170,
      "pname": "Aperol",
      "cname": null,
      "id_category": 4933809643652539,
      "barcode": null,
      "ean13": null,
      "unit": 0,
      "qty_cnt": 1,
      "qty_sum": 3,
      "qty_avg": 3,
      "date_updated": 1641489385000,
      "pp_avg": 10000,
      "pp_unit": 10000
    },
    {
      "id_c": 111111,
      "id_warehouse": 953221254427235,
      "wname": "WAREHOUSE MAIN 21",
      "id_product": 4305305643161675,
      "pname": "01.Vstup 8225",
      "cname": "ACesty",
      "id_category": 5723345161520367,
      "barcode": null,
      "ean13": null,
      "unit": 0,
      "qty_cnt": 1,
      "qty_sum": 6,
      "qty_avg": 6,
      "date_updated": 1642004221000,
      "pp_avg": 100000,
      "pp_unit": 100000
    }
  ],
  "source": "query"
```

```
}
```

POST customers

Slouží k vytvoření nebo aktualizaci objektu zákazníka podle primárního klíče - pokud vytváříte objekt. neposkytujte ID objektu, bude automaticky vytvořeno. Schéma entity je popsána v zod níže.

```
curl 'https://m6vadtaz1h.execute-api.eu-west-1.amazonaws.com/prod/data/customers' \
  -H 'Accept: application/json' \
  -H 'Authorization: A|....' \
  -H 'Content-Type: application/json' \
  --data-raw '{...}'
```

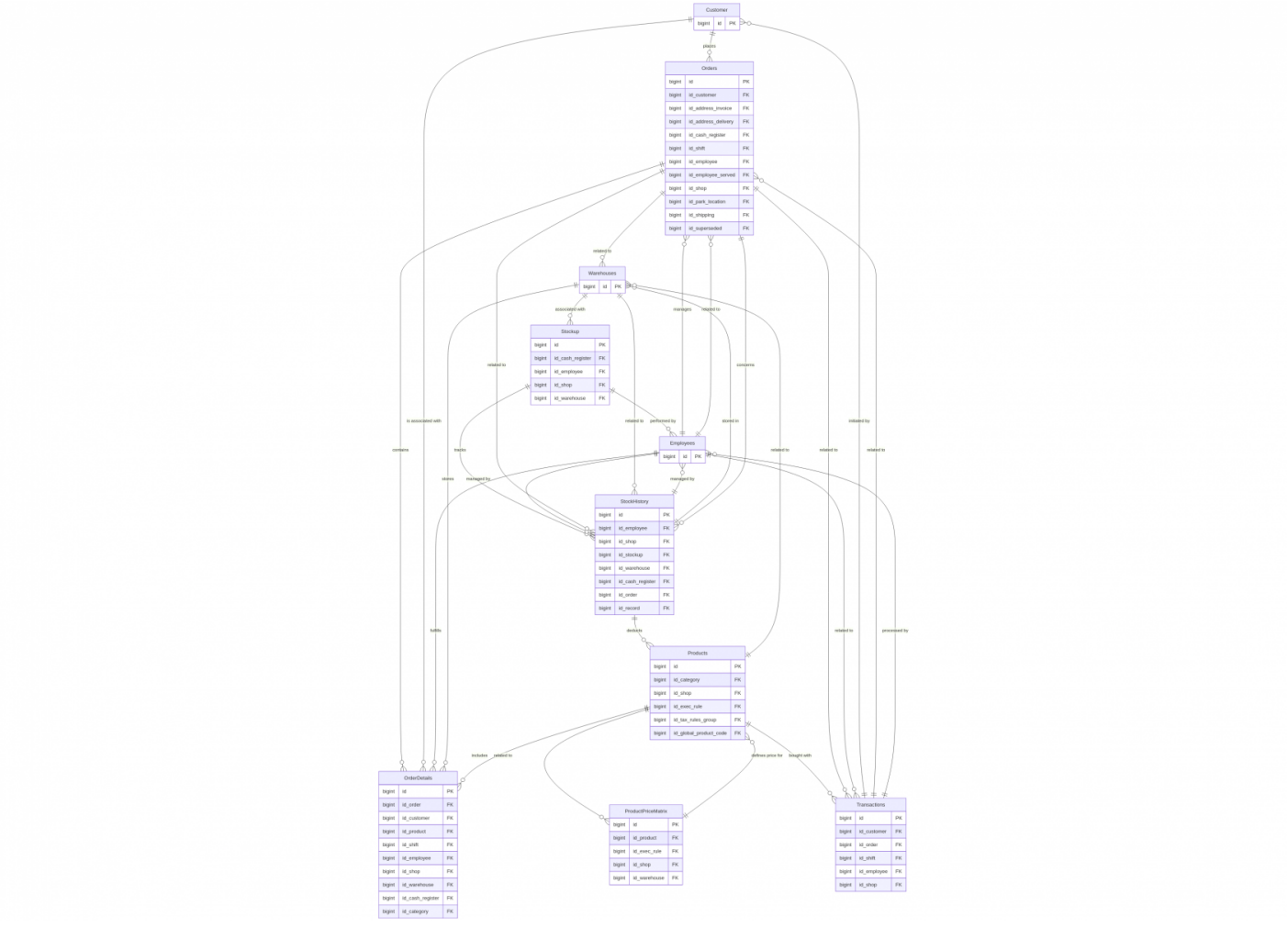
Zasílání objednávek do pokladny

Připravujeme propojení pro zasílání externích objednávek do pokladního systému přes API.

Kontaktujre prosím podpora@kasafik.cz pro více informací.

Vazby - Mermaid

Schema vazeb ve formátu Mermaid



```
erDiagram
    Customer {
        bigint id PK
    }

    Orders {
        bigint id PK
        bigint id_customer FK
        bigint id_address_invoice FK
        bigint id_address_delivery FK
        bigint id_cash_register FK
        bigint id_shift FK
        bigint id_employee FK
        bigint id_employee_served FK
        bigint id_shop FK
        bigint id_park_location FK
        bigint id_shipping FK
        bigint id_superseded FK
    }

    OrderDetails {
        bigint id PK
        bigint id_order FK
        bigint id_customer FK
        bigint id_product FK
        bigint id_shift FK
        bigint id_employee FK
        bigint id_shop FK
        bigint id_warehouse FK
        bigint id_cash_register FK
        bigint id_category FK
    }

    Employees {
        bigint id PK
    }

    StockHistory {
        bigint id PK
        bigint id_employee FK
```

```
    bigint id_shop FK
    bigint id_stockup FK
    bigint id_warehouse FK
    bigint id_cash_register FK
    bigint id_order FK
    bigint id_record FK
}

Transactions {
    bigint id PK
    bigint id_customer FK
    bigint id_order FK
    bigint id_shift FK
    bigint id_employee FK
    bigint id_shop FK
}

Stockup {
    bigint id PK
    bigint id_cash_register FK
    bigint id_employee FK
    bigint id_shop FK
    bigint id_warehouse FK
}

Warehouses {
    bigint id PK
}

Products {
    bigint id PK
    bigint id_category FK
    bigint id_shop FK
    bigint id_exec_rule FK
    bigint id_tax_rules_group FK
    bigint id_global_product_code FK
}

ProductPriceMatrix {
    bigint id PK
```

```
        bigint id_product FK
        bigint id_exec_rule FK
        bigint id_shop FK
        bigint id_warehouse FK
    }

    Customer ||--o{ Orders : places
    Orders ||--o{ OrderDetails : contains
    Customer ||--o{ OrderDetails : "is associated with"
    Employees ||--o{ Orders : "manages"
    Employees ||--o{ OrderDetails : "fulfills"
    Warehouses ||--o{ OrderDetails : "stores"
    Products ||--o{ OrderDetails : "includes"
    StockHistory ||--o{ Employees : "managed by"
    StockHistory ||--o{ Warehouses : "stored in"
    StockHistory ||--o{ Products : "deducts"
    Transactions ||--o{ Customer : "initiated by"
    Transactions ||--o{ Orders : "related to"
    Transactions ||--o{ Employees : "processed by"
    Stockup ||--o{ Employees : "performed by"
    ProductPriceMatrix ||--o{ Products : "defines price for"
    Orders ||--o{ Warehouses : "related to"
    Products ||--o{ Warehouses : "related to"
    Products ||--o{ ProductPriceMatrix : "related to"
    Products ||--o{ Transactions : "bought with"
    Orders ||--o{ Transactions : "related to"
    StockHistory }o--|| Orders : "concerns"
    Warehouses ||--o{ Stockup : "associated with"
    Stockup ||--o{ StockHistory : "tracks"
    Employees ||--o{ StockHistory : "managed by"
    Orders ||--o{ StockHistory : "related to"
    Warehouses ||--o{ StockHistory : "related to"
    Employees ||--o{ Transactions : "related to"
    Employees ||--o{ Orders : "related to"
```

Definice objektu - zod schema

```
const NOT_DELETED = 0;


const bigintOrNumber = z.union([
```

```
    z.number().refine(value => !isNaN(value), {
        message: 'Expected a valid number',
    }),
    z.bigint(),
]);

const BaseSchema = z.object({
    id: z.number(),
    _v: z.number().min(0).default(() => new Date().getTime()),
    _d: z.number().min(0).default(NOT_DELETED),
    date_created: z.number().default(() => new Date().getTime())
});

const CustomerSchema = BaseSchema.extend({
    id: z.bigint().default(BigInt(dbhelpers.getRandomRowId(2))),
    _t: z.literal("customers"),
    firstname: z.string().nullable().optional(),
    customer_type: z.number().int().min(0).default(0), // Assuming customerType should be non-negative integer
    date_updated: z.number().default(() => new Date().getTime()), // Defaults to current date
    color: z.number().int().min(0).default(0), // Assuming color should be non-negative integer
    id_discount_group: bigintOrNumber.optional(),
    zip_code: z.string().nullable().optional(),
    date_birthday: z.number().nullable().optional(),
    geohash: z.string().nullable().optional(),
    company: z.string().nullable(), // Can be null
    company_ico: z.string().nullable().optional(),
    company_dic: z.string().nullable().optional(),
    street: z.string().nullable().optional(),
    city: z.string().nullable().optional(),
    bank_account: z.string().nullable().optional(),
    credit_account: z.string().nullable().optional(),
    internal_json: z.string().nullable().optional(),
    bitmask: z.number().int().min(0).default(0), // Assuming bitmask should be non-negative integer
    barcode: z.string().nullable().optional(),
    email: z.string().email().nullable().optional(), // Assuming email should be a valid email format
    visible: z.boolean().default(true), // Assuming default visibility is true
    lastname: z.string().nullable().optional(),
```

```
    tags: z.string().nullable().optional(),
    country_code: z.string().min(2).max(2).default('CZ'), // Assuming country code is always 2
characters
    price_group: z.enum(['A', 'B', 'C']).default('A'),
    note_external: z.string().nullable().optional(),
    note_internal: z.string().nullable().optional(),
    phone_number: z.string().nullable().optional(),
    date_expires: z.number().nullable().optional(),
    internal_extra: z.string().nullable().optional(),
    price_discount: z.number().int().default(0) // Assuming price discount is an integer
});

const OrdersSchema = BaseSchema.extend({
    id: z.bigint().default(BigInt(dbhelpers.getRandomRowId(7))),
    _t: z.literal("orders"),
    id_address_invoice: bigintOrNumber.optional(),
    id_canceled_order: bigintOrNumber.optional(),
    id_customer: bigintOrNumber.optional(),
    total_points: z.number().default(0),
    total_credits: z.number().default(0),
    total_profit_tax_incl: z.number().default(0),
    total_tax_gst: z.number().default(0),
    dine_in: z.boolean().default(false), // boolean, default to false
    id_tax_report: bigintOrNumber.optional(),
    short_code: z.string().nullable().optional(), // string or null
    date_closed: z.number().nullable().optional(), // number (Date) or null
    date_paid: z.number().nullable().optional(), // number (Date) or null
    total_shipping_tax_incl: z.number().default(0),
    geohash: z.string().nullable().optional(), // string or null
    id_shop: bigintOrNumber,
    total_profit_tax_excl: z.number().default(0),
    total_tax_service: z.number().default(0),
    invoice_number: z.number().nullable().optional(),
    total_discounts_tax_incl: z.number().default(0),
    order_serial_number: z.number().nullable().optional(),
    id_address_delivery: bigintOrNumber.optional(),
    total_shipping_tax_excl: z.number().default(0),
    total_tax_pst: z.number().default(0),
    total_tax_luxury: z.number().default(0),
    date_canceled: z.number().nullable().optional(), // number (Date) or null
    id_cash_register: bigintOrNumber,
```

```
    total_paid_real: z.number().default(0),
    id_shift: bigintOrNumber,
    total_paid_tax_incl: z.number().default(0),
    total_people_seated: z.number().nullable().optional(), // short
    total_tip: z.number().default(0),
    total_discounts_tax_excl: z.number().default(0),
    date_fiscalized: z.number().nullable().optional(), // number (Date) or null
    note: z.string().nullable().optional(), // string or null
    total_paid_tax_excl: z.number().default(0),
    total_wrapping_tax_incl: z.number().default(0),
    id_shipping: bigintOrNumber.optional(),
    id_superseded: bigintOrNumber.optional(),
    total_wrapping_tax_excl: z.number().default(0),
    id_employee: bigintOrNumber,
    currency: z.string().nullable(), // string or null
    conversion_rate: z.number().default(1.0), // number, default to 1.0
    coupon_code: z.string().nullable().optional(), // string or null
    coupon_source: z.string().nullable().optional(), // string or null
    id_payment: z.number().default(0),
    date_tax_reported: z.number().nullable().optional(), // number (Date) or null
    total_tax_eco: z.number().default(0),
    shipping_number: z.string().nullable().optional(), // string or null
    customer_zip: z.string().nullable().optional(), // string or null
    date_loc_acq: z.number().nullable().optional(), // number (Date) or null
    id_employee_served: bigintOrNumber,
    total_products: z.number().default(0),
    id_park_location: bigintOrNumber,
    date_collected: z.number().nullable().optional(), // number (Date) or null
    date_due_pay: z.number().nullable().optional(), // number (Date) or null
    date_emailed: z.number().nullable().optional(), // number (Date) or null
    summary: z.string().nullable().optional(), // string or null
    tags: z.string().nullable().optional(), // string or null
});

// Create a Zod schema for OrderDetails
const OrderDetailsSchema = BaseSchema.extend({
    id: z.bigint().default(BigInt(dbhelpers.getRandomRowId(6))),
    _t: z.literal("order_details"),
    id_payment: z.number().int().default(0), // Default value from TransactionsModel
    id_customer: bigintOrNumber.optional(),
    total_points: z.number().min(0).default(0), // Always whole number, no fractional
```

```
    total_credits: z.number().default(0), // We have an issue, in Orders this is int and here
double
    id_order: bigintOrNumber,
    date_paid: z.number().nullable().optional(),
    id_shop: bigintOrNumber,
    product_barcode: z.string().nullable().optional(),
    total_tax_pst: z.number().int().default(0), // TODO pro prodej poukazu -100Kc je negativni
dan
    total_tax_luxury: z.number().int().min(0).default(0),
    date_canceled: z.number().nullable().optional(),
    id_cash_register: bigintOrNumber,
    product_name: z.string().default("Unknown"),
    product_unit: z.number().int().min(0).default(0),
    tags: z.string().nullable().optional(),
    id_shift: bigintOrNumber,
    product_price_original_tax_excl: z.number().int().default(0),
    id_product: bigintOrNumber,
    product_purchase_price_tax_excl: z.number().nullable().optional(), // Integer or null
    product_attribute_id: z.number().nullable().optional(),
    date_timing_started: z.number().nullable().optional(),
    note: z.string().nullable().optional(),
    date_updated: z.number().nullable().optional(),
    product_upc: z.string().nullable().optional(),
    date_item_printed: z.number().nullable().optional(),
    id_category: bigintOrNumber,
    order_number: z.number().int().min(0).default(0),
    reduction_amount_tax_incl: z.number().int().default(0),
    product_quantity: z.number().default(0), // Assuming whole number as per the comment
    total_weight: z.number().nullable().optional(), // Double or null
    id_employee: bigintOrNumber,
    reference_code: z.string().nullable().optional(),
    currency: z.string().min(3).max(3), // Assuming currency is a 3-character code
    product_price_billed_tax_excl: z.number().int().default(0),
    product_name_alternative: z.string().default(""),
    tax_pst_rate: z.number().default(1.000),
    product_price_billed_tax_incl: z.number().int().default(0),
    reduction_percent: z.number().int().min(0).default(0),
    product_ean13: z.string().nullable().optional(),
    total_tax_eco: z.number().int().default(0),
    product_bitmask: z.number().int().min(0).default(0),
    id_order_details_parent: bigintOrNumber.optional(),
```

```
    id_warehouse: bigintOrNumber,
    date_collected: z.number().nullable().optional(),
    date_voided: z.number().nullable().optional(),
    reduction_amount_tax_excl: z.number().int().default(0),
    product_item_type: z.number().int().min(0).default(0),
});

const EmployeesSchema = BaseSchema.extend({
    id: z.bigint().default(BigInt(dbhelpers.getRandomRowId(4))),
    _t: z.literal("employees"),
    salt: z.string().nullable().optional(),
    visible: z.boolean().default(true),
    pin_require: z.boolean().default(false),
    perm_inventory_bitmask: z.number().int().default(0),
    pin_hash: z.string().nullable().optional(),
    pass_hash: z.string().nullable().optional(),
    perm_cloud_bitmask: z.number().int().default(0),
    tags: z.string().nullable().optional(),
    perm_bitmask: z.number().int().default(0),
    name: z.string(),
    phone_number: z.string().nullable().optional(),
    date_expires: z.number().nullable().optional(), // Stored as milliseconds since epoch
(UNIX timestamp)
    barcode: z.string().nullable().optional(),
    visible_cash_register: z.boolean().default(true),
    otp_hash: z.string().nullable().optional(),
    email: z.string().nullable().optional(),
});

const StockHistorySchema = BaseSchema.extend({
    id: z.bigint().default(BigInt(dbhelpers.getRandomRowId(16))),
    _t: z.literal("stock_history"),
    id_employee: bigintOrNumber,
    id_shop: bigintOrNumber,
    id_stockup: bigintOrNumber.optional().nullable(),
    id_warehouse: bigintOrNumber,
    note: z.string().nullable().optional(),
    id_record: bigintOrNumber,
    date_stocked: z.number().nullable().optional(), // Date as timestamp
    quanity: z.number().default(0.0),
    id_cash_register: bigintOrNumber,
```

```
    id_supplier: bigintOrNumber.optional().nullable(),
    stock_type: z.number().int(),
    id_order: bigintOrNumber.optional().nullable(),
    stock_history_type: z.number().int(),
    supplier_sku: z.string().nullable().optional(),
    product_purchase_unit_price_tax_excl: z.number().int().nullable().optional(),

});

const TransactionsSchema = BaseSchema.extend({
    id: z.bigint().default(BigInt(dbhelpers.getRandomRowId(22))),
    _t: z.literal("transactions"),
    note: z.string().nullable().optional(),
    transaction_code: z.string().nullable().optional(),
    id_customer: bigintOrNumber.optional(),
    transaction_type: z.number().int(),
    id_cash_register: bigintOrNumber.default(BigInt(0)),
    id_order: bigintOrNumber.optional(),
    id_shift: bigintOrNumber.default(BigInt(0)),
    payment_type: z.number().int().min(0),
    total_amount: z.number().int().default(0),
    id_employee: bigintOrNumber,
    id_shop: bigintOrNumber,
    reference_code: z.string().nullable().optional(),
});

const StockupSchema = BaseSchema.extend({
    id: z.bigint().default(BigInt(dbhelpers.getRandomRowId(18))),
    _t: z.literal("stockup"),
    note: z.string().nullable().optional(),
    total_product_purchase_unit_price_tax_excl: z.number().int().nullable().optional(),
    date_stocked: z.number().nullable().optional(), // Assuming timestamp as milliseconds
    id_cash_register: z.number().int(),
    id_supplier: z.number().int().nullable().optional(),
    stock_history_type: z.number().int(),
    id_employee: z.number().int(),
    id_shop: z.number().int(),
    id_warehouse: z.number().int(),
    invoice_number: z.string().nullable().optional(),
});
```

```
const WarehousesSchema = BaseSchema.extend({
    id: z.bigint().default(BigInt(dbhelpers.getRandomRowId(25))),
    _t: z.literal("warehouses"),
    shared: z.boolean().default(false),
    visible: z.boolean().default(true),
    warehouse_bitmask: z.number().int().default(0),
    name: z.string(),
});

const ProductsSchema = BaseSchema.extend({
    id: z.bigint().default(BigInt(dbhelpers.getRandomRowId(11))),
    _t: z.literal("products"),
    image_dimension: z.number().int().nullable().optional(),
    notes_quick: z.string().nullable().optional(),
    quantity_multiple: z.number().default(1),
    quantity_minimal: z.number().default(1),
    additional_shipping_cost: z.number().default(0),
    date_updated: z.number(), // Date as timestamp (milliseconds)
    inventory_management: z.boolean().default(false),
    color: z.number().int().default(0),
    item_type: z.number().int().default(0),
    id_category: bigintOrNumber.default(BigInt(0)),
    tax_luxury: z.number().default(1.000),
    ean13: z.string().nullable().optional(),
    id_tax_rules_group: bigintOrNumber.nullable().default(BigInt(0)),
    name_alternative: z.string().nullable().optional(),
    sku: z.string().nullable().optional(),
    tax_eco: z.number().default(1.000000),
    barcode: z.string().nullable().optional(),
    height: z.number().nullable().optional(),
    visible: z.boolean().default(true),
    image_url: z.string().nullable().optional(),
    warranty_length: z.number().int().default(0),
    condition_type: z.number().int().default(0),
    tax_pst_rate: z.number().default(1.000),
    upc: z.string().nullable().optional(),
    weight: z.number().nullable().optional(), // Stored as alcoholTax in Java
    icon_code: z.number().int().default(0),
    details_url: z.string().nullable().optional(),
    warranty_type: z.number().int().default(0),
    tags: z.string().nullable().optional(),
```

```
    description_short: z.string().nullable().optional(),
    unit: z.number().int().default(0),
    id_exec_rule: bigintOrNumber.default(BigInt(0)),
    depth: z.number().nullable().optional(),
    sale_group_type: z.number().int().default(0),
    id_global_product_code: bigintOrNumber.nullable().optional(),
    accounting_group: z.number().int().default(600000),
    name: z.string(),
    width: z.number().nullable().optional(),
    node_sort: bigintOrNumber.default(BigInt(0)),
    attributes_bitmask: z.number().int().default(0),
    inventory_type: z.number().int().default(0),
    internal_extra: z.string().nullable().optional(),
    id_shop: bigintOrNumber.default(BigInt(0))
});

const ProductPriceMatrixSchema = BaseSchema.extend({
    id: z.bigint().default(BigInt(dbhelpers.getRandomRowId(12))),
    _t: z.literal("product_price_matrix"),
    unit_price_base_tax_excl: z.number().nullable().optional(), // decimal value 19.95
    unit_price_base_tax_incl: z.number().nullable().optional(), // decimal value 22.99
    date_starts: z.number(), // Date as timestamp (milliseconds)
    unit_price_regular_tax_excl: z.number().int().default(0),
    unit_price_c_tax_excl: z.number().int().nullable().optional(),
    points_deduct: z.number().default(0.0),
    credits_deduct: z.number().default(0.0),
    hour_bitmask: z.number().int().default(0),
    reduction_amount_a_tax_excl: z.number().int().default(0),
    id_product: bigintOrNumber,
    id_exec_rule: bigintOrNumber.default(BigInt(0)),
    unit_price_b_tax_excl: z.number().int().nullable().optional(),
    price_bitmask: z.number().int().default(0),
    dow_bitmask: z.number().int().default(0),
    id_shop: bigintOrNumber.default(BigInt(0)),
    currency: z.string(),
    date_expires: z.number().nullable().optional(), // Date as timestamp (milliseconds)
    unit_price_a_tax_excl: z.number().int().nullable().optional(),
    id_warehouse: bigintOrNumber.default(BigInt(0)),
});
```

Revision #67
Created 2022-01-12 13:28:04 UTC by Admin
Updated 2026-06-04 20:40:32 UTC by Admin